ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - Tj141502

INTERNET OF THINGS UNTUK MEMANTAU KONDISI SERTA AKTIVITAS MANULA MENGGUNAKAN TURTLEBOT SEBAGAI GATEWAY

Anak Agung Ngurah Surya Laksamana
NRP 07211440000050

Dosen Pembimbing
Muhtadin, ST., MT.
Arief Kurniawan, ST., MT.

DEPARTEMEN TEKNIK KOMPUTER
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – TJ141502

INTERNET OF THINGS UNTUK MEMANTAU KONDISI SERTA AKTIVITAS MANULA MENGGUNAKAN TURTLEBOT SEBAGAI *GATEWAY*

Anak Agung Ngurah Surya Laksamana
NRP 07211440000050

Dosen Pembimbing
Muhtadin, ST., MT.
Arief Kurniawan, ST., MT.

DEPARTEMEN TEKNIK KOMPUTER
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2018

FINAL PROJECT – TJ141502

INTERNET OF THINGS FOR ELDERLY CONDITION AND ACTIVITY MONITORING USING TURTLEBOT AS A GATEWAY

Anak Agung Ngurah Surya Laksamana
NRP 07211440000050

Advisor
Muhtadin, ST., MT.
Arief Kurniawan, ST., MT.

Department of Computer Engineering
Faculty of Electrical Technology
Sepuluh Nopember Institute of Technology
Surabaya 2018

PERNYATAAN KEASLIAN TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "***Internet of Things* untuk Memantau Kondisi serta Aktivitas Manula Menggunakan TurtleBot sebagai *Gateway***" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 27 Desember 2017

Anak Agung Ngurah Surya L

NRP. 07211440000050

LEMBAR PENGESAHAN

Internet of Things* untuk Memantau Kondisi serta Aktivitas Manula Menggunakan *TurtleBot* sebagai *Gateway

Tugas Akhir ini disusun untuk memenuhi salah satu syarat memperoleh gelar Sarjana Teknik di Institut Teknologi Sepuluh Nopember Surabaya

Oleh : Anak Agung Ngurah Surya L (NRP: 07211440000050)

Tanggal Ujian : 4 Januari 2018 Periode Wisuda : Maret 2018

Disetujui oleh:

(Pembimbing I)

Muhtadin, ST., M.Sc.
NIP: 198106092009121003 .....

(Pembimbing II)

Arief Kurniawan, ST., MT.
NIP: 197409072002121001 .....

(Penguji I)

Dr. I Ketut Eddy Purnama, ST., MT.
NIP: 196907301995121001 .....

(Penguji II)

Dr. Diah Puspito Wulandari, ST., M.Sc.
NIP: 198012192005012001 .....

(Penguji III)

Ahmad Zaini, ST., M.Sc.
NIP: 197504192002121003 .....

Mengetahui
Kepala Departemen Teknik Komputer

Dr. I Ketut Eddy Purnama, ST., MT.
NIP: 196907301995121001

ABSTRAK

Nama Mahasiswa	:	Anak Agung Ngurah Surya Laksamana
Judul Tugas Akhir	:	*Internet of Things* untuk Memantau Kondisi serta Aktivitas Manula Menggunakan TurtleBot sebagai *Gateway*.
Pembimbing	:	1. Muhtadin, ST., MT. 2. Arief Kurniawan, ST., MT.

Dalam beberapa dekade terakhir, jumlah populasi manula meningkat secara signifikan. Oleh karena itu, banyak manula yang harus dipantau oleh keluarganya. Sayangnya, tidak semua anggota keluarga dapat memerhatikan mereka secara langsung karena beberapa alasan seperti pekerjaan atau sekolah. Maka dari itu dibutuhkan sebuah sistem yang dapat memantau kondisi manula dari jarak jauh dengan sistem yang terintegrasi antara *robot* dan *wearable device*. Pada tugas akhir ini ditunjukkan bahwa sistem pemantauan dapat digunakan dalam kehidupan untuk memantau kondisi manula. Sistem dapat digunakan baik oleh keluarga ataupun dokter unruk memantau kondisi melalui jaringan internet. Disamping pemantauan, sistem bisa melakukan pencegahan dini tentang kondisi tidak diinginkan yang mungkin terjadi pada manula. Sistem memiliki dua komponen utama, *wearable device* (*device* yang digunakan untuk menganalisa kondisi atau sebagai *trigger*) dan *robot*. *Wearable device* dilengkapi beberapa sensor yang dapat menganalisa kondisi vital pada manula. Tugas utama robot adalah datang menuju manula ketika *wearable device* ter-*trigger* dan mengambil gambar dan kemudian mengirimnya sebagai sebuah notifikasi kepada keluarga manula melalui *messenger*. Sistem pendeteksi jatuh dapat mendeteksi kondisi jatuh dari pengguna *wearable device* dan mengirimnya melalui *messenger* melalui personal *chat* atau grup.

Kata Kunci : Aktivitas, Kondisi, Manula, *Robot*, *Wearable Device*.

Halaman ini sengaja dikosongkan

ABSTRACT

Name	:	Anak Agung Ngurah Surya Laksamana
Title	:	*Internet of Things for Eldery Condition and Activity Monitoring Using TurtleBot as a Gateway*
Advisors	:	1. Muhtadin, ST., MT. 2. Arief Kurniawan, ST., MT.

In recent years, the number of older people is increasing rapidly. Therefore, a lot of elderly have to be monitored by their family. Unfortunately, not all of the family member are available to pay attention for them directly because of a lot of reason like work or study. Hence, we propose a system that can be used to monitor the elderly status remotely with an integrated system between robot and wearable devices. This research demonstrated that this system can be used in the real life and in the future as a monitoring system to the elderly people. It can be used by either family as a personal usage or doctor from to monitor elderly condition through the internet connection. Besides monitoring, the system can do some early prevention about risky situation that may be occurring in elder condition. The system originally has two main components, the wearable (device to analyze the condition or as a trigger) and the robot. The wearable device is equipped by some sensors that can analyze the humans vital sign as a condition for elderly. The main function of the robot is to come to elder whenever the critical event is triggered and taking the picture of it in panorama view than send it as a notification for the family through messenger system. Our fall detector system can detect the fall event from volunteers wearable device and send the notification through a messenger as a personal chat or group chat.

Keywords :Activity, Condition, Elderly, Robot, Wearable Device.

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Puji dan syukur kehadirat Tuhan Yang Maha Esa atas segala karunia-Nya, penulis dapat menyelesaikan Tugas Akhir ini dengan judul ***Internet of Things* Untuk Memantau Kondisi Serta Aktivitas Manula Menggunakan TurtleBot Sebagai *Gateway***.

Tugas Akhir ini disusun dalam rangka pemenuhan bidang riset di Departemen Teknik Komputer ITS serta digunakan sebagai persyaratan menyelesaikan pendidikan Sarjana. Tugas Akhir ini dapat terselesaikan tidak lepas dari bantuan berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. Keluarga, Ayah dan Ibu yang telah memberikan dorongan spiritual dan material dalam penyelesaian Buku Tugas Akhir ini.
2. Bapak Kepala Departemen Teknik Komputer Dr. I Ketut, ST., MT.
3. Bapak Muhtadin, ST., M.Sc. dan Bapak Arief Kurniawan, ST., MT. atas bimbingan selama mengerjakan Tugas Akhir.
4. Bapak - Ibu Dosen pengajar Departemen Teknik Komputer atas pengajaran, bimbingan, serta perhatian yang diberikan kepada penulis selama ini.
5. Naisha Sebby Alkafilah yang selalu memberi dukungan moral dari awal pengerjaan hingga penulis menyelesaikan Tugas Akhir ini.
6. Rahadian Esa Galang yang membantu penulis dalam melakukan pengujian.
7. Seluruh teman - teman B401 Laboratorium Komputasi Multimedia yang sedikit banyak membantu menyelesaikan buku ini.

Kesempurnaan hanya milik Tuhan, untuk itu penulis memohon segenap kritik dan saran yang membangun. Semoga Tugas Akhir ini dapat memberikan manfaat bagi kita semua. Amin.

Surabaya, 10 November 2017

Penulis

DAFTAR ISI

Halaman ini sengaja dikosongkan

DAFTAR GAMBAR

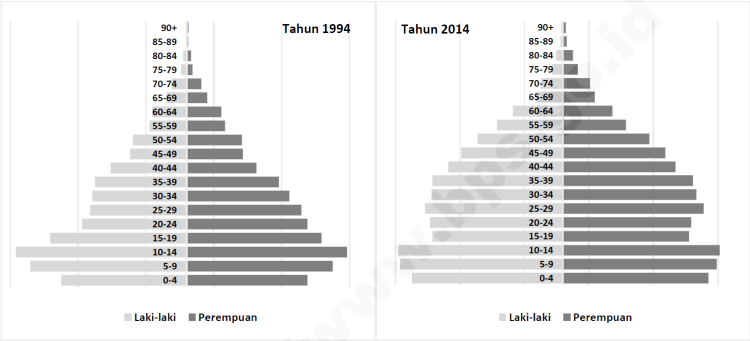
DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB 1
PENDAHULUAN

1.1 Latar belakang

Manusia lanjut usia (manula) merupakan tahap lanjut dari suatu proses kehidupan manusia yang ditandai dengan penurunan kemampuan tubuh untuk beradaptasi dengan lingkungan. Menurunnya kemampuan tubuh untuk beradaptasi ini membuat manula rentan mengalami kecelakaan fisik seperti terjatuh. Dukungan dan pengawasan dari keluarga terhadap manula sangat diperlukan, agar terhindar dari hal-hal yang tidak diinginkan. Namun, seiring dengan bertambahnya usia, seseorang memiliki tanggung jawab yang lebih besar sehingga tidak dapat memantau dan mengawasi keluarga terdekatnya (manula) terlebih jika dalam keluarga itu memiliki sedikit anggota keluarga.



Gambar 1.1: Piramida penduduk Indonesia tahun 1994 dan 2014 [1]

Dari kedua piramida tersebut terlihat pula bahwa ujung piramida, yaitu dimulai dari kelompok usia 60 tahun ke atas, semakin melebar berarti terjadi peningkatan penduduk lansia.Seperti yang dijelaskan pada [1] penurunan angka kelahiran, peningkatan angka harapan hidup, dan bertambahnya jumlah penduduk lansia dari tahun ke tahun menunjukkan bahwa struktur penduduk Indonesia bertransisi ke arah struktur penduduk tua *(ageing population)* sehingga rasio ketergantungan lansia Indonesia pada tahun 2014 se-

besar 12,71:100 . Artinya bahwa setiap 100 orang penduduk usia produktif harus menanggung sekitar 13 orang lansia.

Berdasarkan data Susenas 2014 seperti yang dijelaskan pada [1], jumlah rumah tangga lansia sebanyak 16,08 juta rumah tangga atau 24,50 persen dari seluruh rumah tangga di Indonesia. Rumah tangga lansia adalah yang minimal salah satu anggota rumah tangganya berumur 60 tahun ke atas. Jumlah lansia di Indonesia mencapai 20,24 juta jiwa, setara dengan 8,03 persen dari seluruh penduduk Indonesia tahun 2014.Sebagian besar lansia di Indonesia masih tinggal dalam satu rumah tangga bersama dengan keluarga besarnya. Sebesar 42,32 persen lansia tinggal bersama tiga generasi dalam satu rumah tangga dan sebesar 26,80 persen lansia tinggal bersama keluarga inti dalam satu rumah tangga. Hanya sekitar 10 persen lansia yang tinggal sendiri dan 17,48 persen yang tinggal bersama pasangannya.

Seiring dengan itu pula, banyak teknologi yang bisa digunakan untuk membantu manusia dalam mengatasi hal tersebut. Seperti telah berkembangnya sensor-sensor untuk mendeteksi berbagai perubahan, teknologi otomasi robot, perangkat pintar, dan masih banyak lagi. Teknologi-teknologi ini dapat diintegrasikan untuk melakukan suatu fungsi tertentu yang bertujuan untuk mempermudah kehidupan manusia seperti pengingat, pengawasan, pemberitahuan dan lainnya.

Berdasarkan uraian latar belakang diatas dibuatlah sebuah sistem yang mengintegrasikan robot dengan sebuah *wearable device* untuk melakukan pemantauan terhadap kondisi manula sehingga diharapkan anggota keluarga dapat memantau anggota keluarga mereka yang berstatus manula dari jarak jauh.

1.2 Permasalahan

Berdasarkan data yang telah dipaparkan pada latar belakang, dapat dirumuskan permasalahan sebagai berikut:

1. Sibuknya seorang anggota keluarga membuat pengawasan terhadap manula dalam keluarga itu menjadi minim.
2. Belum adanya suatu sistem di Indonesia yang bisa memantau kondisi manula dan memberitahukan kerabat terdekat dan dokter jika terjadi sesuatu pada kondisi manula melalui *mes-*

senger.

3. Meningkatnya populasi lansia dalam periode waktu tertentu membuat sistem pengawasan terhadap lansia sangat dibutuhkan.

1.3 Tujuan

Tujuan dari penelitian ini adalah membuat suatu sistem purwarupa untuk memantau keadaan manula serta memberi peringatan jika kondisi manula dalam keadaan bahaya menggunakan *microcontroller* dan turtlebot [1], sehingga nantinya sistem ini dapat digunakan untuk orang-orang yang tidak bisa memantau anggota keluarganya yang berstatus manula secara langsung.

1.4 Batasan masalah

Untuk memfokuskan permasalahan yang akan diangkat maka dilakukan pembatasan masalah. Batasan-batasan masalah tersebut diantaranya adalah:

1. Hanya memonitoring kondisi *vital sign* pada manula, menentukan tingkat aktivitas dan jatuh atau tidaknya manula.
2. Memberikan peringatan melalui aplikasi *messenger* berbasis *mobile*.
3. Turtlebot tidak mengikuti pergerakan manula secara otomatis melainkan menggunakan *path planning* dan berfungsi sebagai *gateway* untuk mengirim dan menerima data.
4. *Wearable device* hanya bekerja pada kondisi normal dan mengabaikan kondisi-kondisi yang dapat merusak alat seperti percikan air dan sebagainya.
5. TurtleBot hanya beroperasi pada permukaan yang datar.

1.5 Sistematika Penulisan

Laporan penelitian Tugas akhir ini tersusun dalam sistematika dan terstruktur sehingga mudah dipahami dan dipelajari oleh pembaca maupun seseorang yang ingin melanjutkan penelitian ini. Alur sistematika penulisan laporan penelitian ini yaitu :

[1] TurtleBot akan dijelaskan lebih mendalam pada bab 2

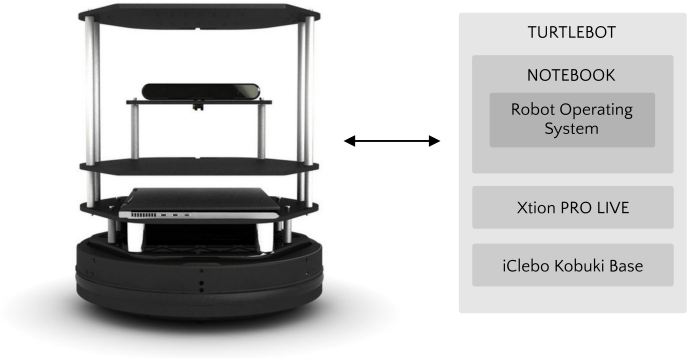
1. BAB I Pendahuluan
   Bab ini berisi uraian tentang latar belakang permasalahan, batasan masalah yang diteliti, tujuan penelitian, dan sistematika laporan.
2. BAB II Dasar Teori
   Pada bab ini berisi tentang uraian sistematis teori penunjang yang menjadi referensi dalam pengerjaan tugas akhir. Teori penunjang tersebut meliputi turtlebot, wearable device, cubeacon dan localtunnel.
3. BAB III Perancangan Sistem dan Impementasi
   Bab ini berisi penjelasan tentang desain atau perancangan sistem yang diwujudkan dalam bentuk blok diagram penelitian dan implementasi sistem yang merupakan pelaksanaan teknis dari setiap blok diagram pada desain sistem. Dalam bab ini juga dijelaskan tentang parameter-parameter yang digunakan dalam penelitian.
4. BAB IV Pengujian dan Analisa
   Bab ini menjelaskan mengenai hasil pengujian dalam memantau kondisi manula melalui turtlebot dan melakukan pencegahan dini yang dapat dilakukan keluarga melalui notifikasi serta analisis terkait tingkat keberhasilan pengujian baik secara keseluruhan maupun secara khusus terkait pengaruh parameter-parameter yang digunakan dalam pengujian.
5. BAB V Penutup
   Bab ini merupakan penutup yang berisi kesimpulan yang diambil dari tugas akhir dan pengujian yang telah dilakukan. Saran dan kritik yang membangun untuk pengembangkan lebih lanjut juga dituliskan pada bab ini.

BAB 2
TINJAUAN PUSTAKA

Demi mendukung Tugas Akhir ini, dibutuhkan beberapa teori penunjang sebagai bahan acuan dan referensi. Dengan demikian Tugas Akhir ini menjadi lebih terarah.

2.1 *TurtleBot*

Saat ini robot tidak hanya digunakan dalam bidang industri saja, namun saat ini robot telah menjadi hal umum dalam kehidupan sehari-hari. Pendidikan, pertanian, perikanan, militer, serta beberapa kegiatan yang mempunyai resiko yang sangat besar telah menggunakan tenaga robot agar lebih efisien dan cepat. TurtleBot merupakan sebuah *robot service open platform* yang dapat digunakan untuk pembelajaran serta penelitian tentang *robot*. Desain turtlebot dapat diubah-ubah sesuai kebutuhan sehingga robot ini tidak terbatas untuk penggunaan tertentu. turtlebot terdiri dari *kobuki base*, *RGBD* sensor, *netbook* atau *Single Board Processor (SBC)*, dan rangka yang digunakan sebagai dudukan komponen lainnya.



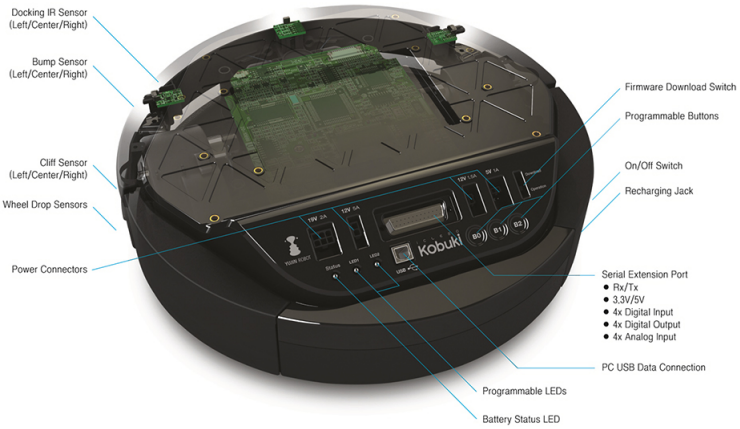
Gambar 2.1: *TurtleBot 2* [2]

2.1.1 *Robot Operating System*

Robot Operating System atau *ROS* adalah sebuah *framework* yang berjalan pada sistem operasi Linux dan OS X yang digunakan untuk membuat program robot yang fleksibel dan mudah untuk digunakan. *ROS* berisi berbagai *tool, library, driver* dan konvensi yang bertujuan untuk mempermudah pembuatan aplikasi atau program pada berbagai *platform robot*. Sehingga *ROS* memungkinkan penggunanya untuk menggunakan *style* yang sama dalam membuat sebuah aplikasi atau robot jenis apapun [8].

2.1.2 *iClebo Kobuki Base*

Iclebo Kobuki merupakan *low-cost mobile research base* yang digunakan untuk pendidikan dan penelitian dalam bidang robotika. Didalam kobuki terdapat 3 buah *Docking IR* Sensor, 3 buah *bumper* sensor, 3 buah *Cliff* sensor, sensor *Gyro*, Motor DC, dan *Battery Lithium-Ion* [2]. Namun *kobuki base* tersebut membutuhkan *netbook* atau *Single Board Computer* untuk membuatnya berfungsi. Terdapat *usb interface* dan *serial port* yang digunakan untuk melakukan komunikasi dari *netbook* atau *SBC* ke *kobuki*.



Gambar 2.2: *iClebo Kobuki Base*

2.1.3 *Netbook* atau *SBC*

Agar turtlebot dapat bekerja dibutuhkan sebuah netbook, pada netbook tersebut dipasang *framework Robot Operating System (ROS)*. *ROS* berisi beragam peralatan (*tools*), *library*, *driver*, dan konvensi yang bertujuan untuk memudahkan pembuatan program pada berbagai *platform robot*. Dengan sistem *ROS* para pengembang yang berbeda-beda dapat bekerja sama mengembangkan suatu karya [8]. Pada *ROS* terdapat *library* dan *driver* yang diperlukan untuk menggerakan *iClebo Kobuki*, kita dapat dengan mudah untuk mengontrol *kobuki base*.

2.2 *Wearable Device*

Wearable Device adalah perangkat yang akan dikenakan oleh lansia. Perangkat ini bertujuan untuk memantau keadaan lansia yang mengenakannya. perangkat ini memiliki beberapa sensor sehingga mampu mendeteksi keadaan tidak normal pada lansia dan mengirimkan sinyal kepada robot untuk diteruskan sebagai notifikasi. Protokol komunikasi yang digunakan oleh perangkat ini adalah *WiFi*. Nantinya, lansia akan mengenakan perangkat ini pada sekitaran pinggang mereka. Kebanyakan pengembang didunia, biasanya mengembangkan perangkat ini menggunakan *microcontroller* sebagai komponen utama dan beberapa sensor untuk mengakuisisi datanya.

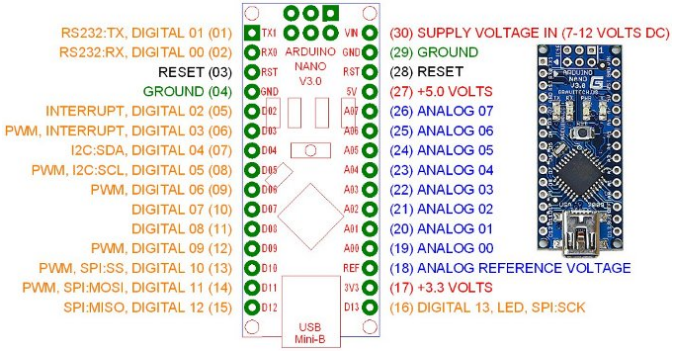
2.2.1 *Arduino Nano*

Arduino merupakan papan mikrokontroler yang dapat melakukan proses *input* maupun *output* dengan cara *upload* program tertentu, sebagai contoh program akuisisi data oleh sensor suhu. Pada Arduino Nano mikrokontroler yang digunakan yakni ATmega328P. Arduino Nano beroperasi pada tegangan lima *volt* dan tegangan yang direkomendasikan sebesar tujuh sampai 12 *volt*.

Gambar 2.3 merupakan *pinout* diagram pada Arduino Nano. Seperti yang dijelaskan pada [9], Arduino Nano memiliki 14 buah digital pin yang dapat digunakan sebagai input atau output, sengan menggunakan fungsi *pinMode()*, *digitalWrite()*, dan *digital(Read)*. Pin-pin tersebut bekerja pada tegangan 5V, dan setiap pin dapat

menyediakan atau menerima arus 20mA, dan memiliki tahanan *pull-up* sekitar 20-50k ohm (secara default dalam posisi *disconnect*). Nilai maksimum adalah 40mA, yang sebisa mungkin dihindari untuk menghindari kerusakan chip mikrokontroler. Beberapa pin memiliki fungsi khusus :

1. **Serial**, terdiri dari 2 pin : pin 0 (RX) dan pin 1 (TX) yang digunakan untuk menerima (RX) dan mengirim (TX) data serial.
2. ***External Interrupts***, yaitu pin 2 dan pin 3. Kedua pin tersebut dapat digunakan untuk mengaktifkan *interrupts*. Gunakan fungsi *attachInterrupt()*.
3. ***Pulse Width Modulation (PWM)***: Pin 3, 5, 6, 9, 10, dan 11 menyediakan output *PWM* 8-bit dengan menggunakan fungsi *analogWrite()*.
4. ***Serial Peripheral Interface (SPI)*** : Pin 10 (SS), 11 (*MOSI*), 12 (*MISO*), dan 13 (*SCK*) mendukung komunikasi *SPI* dengan menggunakan SPI Library
5. ***LED*** : Pin 13. Pada pin 13 terhubung *built-in led* yang dikendalikan oleh digital pin no 13.

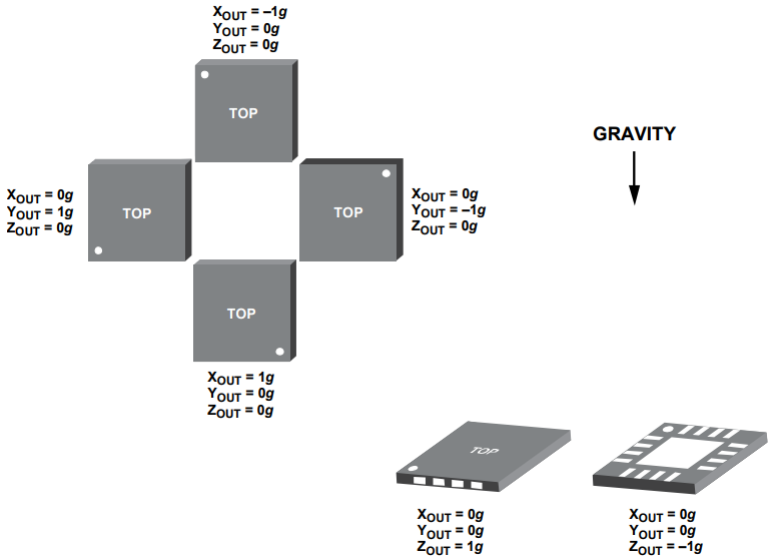


Gambar 2.3: Arduino Uno *pinout*

2.2.2 *ADXL335*

ADXL335 adalah sebuah modul *accelerometer* yang memiliki 3 axis serta bekerja dalam daya yang rendah. Modul ini digu-

nakan untuk mengukur nilai percepatan baik itu percepatan secara static seperti kemiringan dan percepatan dinamis seperti gerakan, guncangan, dan getaran. Dalam penggunaannya, sensor ini harus dikalibrasikan terlebih dahulu agar didapat nilai 0 setiap axis saat posisi modul sedang normal. Setelah dilakukan kalibrasi, modul akan mengeluarkan nilai minus jika salah satu orientasi dari ketiga axis dibalik.



Gambar 2.4: perbandingan antara nilai Respon *Output* dan orientasi gravitasi [3]

2.2.3 *AD8232 ECG Sensor Module*

AD8232 adalah sebuah modul *Electrocardiography (ECG)* untuk memantau kondisi jantung dari penggunanya. Pengambilan *ECG* dilakukan dengan memasangkan tiga elektroda pada bagian tubuh sebagai berikut:

1. Kutub positif bagian dada sebelah kanan
2. Kutub netral bagian dada sebelah kiri

3. Kutub negatif bagian perut sebelah kiri

Data yang diakuisisi oleh elektroda tersebut merupakan tegangan yang muncul saat jantung berkontraksi dan relaksasi. Data yang muncul berupa data analog tegangan yang telah dikonversi antara nol sampai lima volt. Menurut *American Heart Association* pada [4], estimasi detak jantung yang sehat memiliki nilai yang berbeda untuk setiap umur. Estimasi tersebut terlhiat pada gambar 2.5.

Age	Target HR Zone 50-85%	Average Maximum Heart Rate, 100%
20 years	100-170 beats per minute	200 beats per minute
30 years	95-162 beats per minute	190 beats per minute
35 years	93-157 beats per minute	185 beats per minute
40 years	90-153 beats per minute	180 beats per minute
45 years	88-149 beats per minute	175 beats per minute
50 years	85-145 beats per minute	170 beats per minute
55 years	83-140 beats per minute	165 beats per minute
60 years	80-136 beats per minute	160 beats per minute
65 years	78-132 beats per minute	155 beats per minute
70 years	75-128 beats per minute	150 beats per minute

Gambar 2.5: Estimasi denyut jantung per menit yang sehat dalam berbagai umur [4]

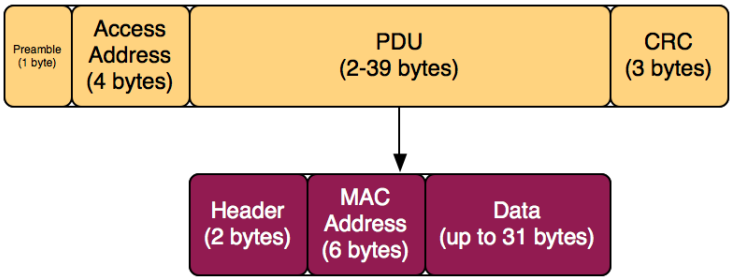
2.2.4 *ESP8266 ESP-01*

ESP8266 ESP-01 adalah sebuah modul *Wi-Fi* yang membuat sebuah *microcontroller* dapat terhubung ke jaringan *Wi-Fi*. Modul ini memiliki sebuah *SOC (System On a Chip)* yang memungkinkan untuk mendapatkan nilai input ataupun output tanpa adanya *microcontroller*. Hal ini terjadi karena ESP-01 bertindak sebagai komputer mini. Walaupun modul ini dapat bekerja tanpa adanya *microcontroller*, modul ini juga bisa memberikan akses sebuah *microcontroller* ke sebuah jaringan *Wi-Fi*. dalam komunikasi datanya, modul ini dapat beroperasi dalam 3 mode yaitu *Access Point (AP),*

Station (STA) atau beroperasi dalam 2 mode sekaligus. Dalam mode *AP*, komunikasi yang terjadi hanyalah 2 arah yaitu antara modul dan perangkat yang terhubung dengan modul. Dalam mode *STA*, modul menjadi perantara antara sebuah *router* dan perangkat yang terkoneksi dengan modul atau bisa disebut sebagai *repeater*.

2.2.5 *Cubeacon*

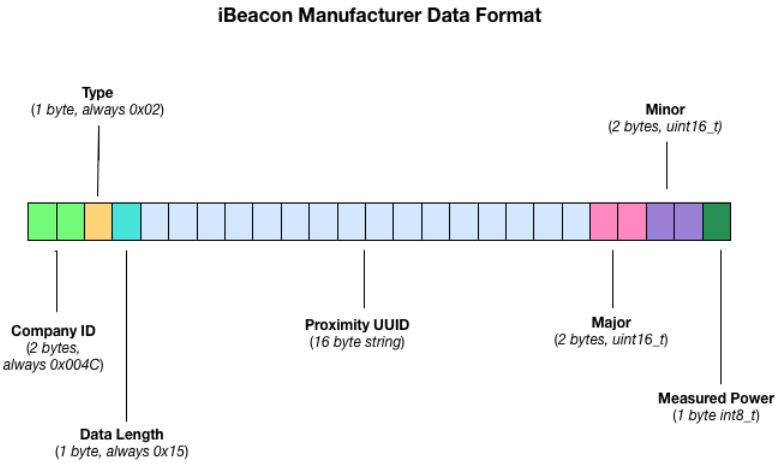
Cubeacon merupakan sebuah *developer kit* yang menggunakan teknologi *iBeacon* milik Apple. Perangkat ini merupakan perangkat yang dapat melakukan *broadcast identifier* atau ID pengenal dari perangkat ke *portable device* terdekat seperti *smartphone* melalui protokol *beacons*, yang merupakan kelas dari *Bluetooth low energy*. *Bluetooth Low Energy* melakukan transmisi data dengan mengirim paket data sebesar 47 *bytes* dimana 1 *byte* merupakan *preamble*, 4 bytes merupakan access address, 2-39 *bytes* merupakan *advertising channel PDU* dan 3 *bytes* terakhir merupakan *CRC*.



Gambar 2.6: Paket data *Bluetooth low energy* [5]

Sesuai dengan penjabaran pada [5] *advertising channel protocol data unit* atau yang selanjutnya disebut *advertising channel PDU* adalah merupakan potongan data pada *layer model advertising*, biasanya terdiri dari 39 *bytes* dimana 2 *bytes* awal merupakan *header*, 6 *bytes* merupakan *MAC address* dan 31 *bytes* terakhir merupakan data. Dalam tugas akhir ini yang perlu diperhatikan adalah 31 *bytes* data. Contohnya, sebuah *iBeacon* mentransmisikan data pada sebuah *smartphone* dan didapat paket data seperti berikut: 02 01

06 1A FF 4C 00 02 15 B9 40 7F 30 F5 F8 46 6E AF F9 25 55 6B 57 FE 6D 00 49 00 0A C5. Kemudian akan dicoba dikelompokkan dalam 3 grup berdasarkan jenis datanya, paket data akan terlihat seperti berikut: [02 01 06 1A FF 4C 00 02 15] [B9 40 7F 30 F5 F8 46 6E AF F9 25 55 6B 57 FE 6D] [00 49] [00 0A] [C5]. Berdsarkan pengelompokkannya, 9 *bytes* pertama merupakan data prefix, 16 *bytes* berikutnya merupakan *proximity UUID*, 2 *bytes major*, 2 *bytes minor* dan 1 *bytes measured power*.



Gambar 2.7: Format Data *iBeacon* [6]

2.3 *If This Then That*

Sesuai dengan situs resminya pada [10], *If This Then That (IFTTT)* adalah merupakan sebuah layanan berbasis web untuk membuat sebuah *conditional statements* sederhana yang disebut *applets*. Sebuah *applet* dapat di-*trigger* dari adanya perubahan pada *web service* lainnya seperti Gmail, Facebook, Instagram dan lain-lain. Sebagai contoh, seorang *user* membuat sebuah *applet* untuk menghubungkan akun instagram dan facebook, sehingga apapun foto yang di *post* oleh user pada instagram akan otomatis muncul pada *timeline* facebook. Hal ini memungkinkan untuk mengkonek-

sikan satu *device* dengan berbagai macam service yang didukung oleh *platform* ini.

Pada tugas akhir ini, *IFTTT* digunakan untuk menghubungkan robot dengan portal pesan LINE. Robot akan menginputkan gambar kepada *IFTTT* kemudian *IFTTT* akan mem-*bundle* informasi tersebut menjadi sebuah notifikasi yang akan diteruskan ke LINE pengguna.

2.4 *Cloud Storage*

Cloud Storage merupakan media penyimpanan secara daring. Saat ini penggunaan *cloud* telah menjadi paradigma baru dalam dunia teknologi informasi seperti *internet of things*. Banyak data yang disimpan di *cloud* agar dapat diakses oleh pengguna dimana pun mereka berada. Saat ini telah banyak penyedia layanan *cloud storage* secara gratis maupun berbayar. Banyak perusahaan digital maupun bukan digital hingga pemerintahan telah menggunakan penyimpanan *cloud* sebagai tempat penyimpanan data-data yang dikelola maupun sebagai *back up* data. Sesuai dengan penjelasan pada [11], dari segi privasi, *cloud storage* dibagi menjadi empat jenis yaitu:

1. *Personal Cloud Storage*
   Storage jenis ini biasanya digunakan oleh perseorangan untuk *back up* data-data yang dimiliki. Beberapa *smartphone* bahkan telah menambahkan aplikasi *cloud* untuk mempermudah penggunanya ketika ingin melakukan *back up* terhadap datanya.
2. *Public Cloud Storage*
   Jenis *storage* ini sering digunakan oleh perusahaan yang memiliki jenis data tidak terstruktur. Perusahaan menyewa *cloud storage* dari *provider cloud* dan meminta untuk mengelola seluruh data yang dimiliki perusahaan. Selain itu jenis *storage* ini sering digunakan dalam hal berbagi secara publik.
3. *Private Cloud Storage*
   Jenis ini umumnya digunakan oleh perusahaan yang menginginkan fasilitas keamanan serta kecepatan lebih dibanding jenis *cloud storage* yang lain. Pada *storage* ini *provider* akan membentuk sebuah infrastruktur dalam pusat data perusaha-

an, dan keduanya akan berintegrasi didalamnya.

4. *Hybrid Cloud Storage*
   Merupakan gabungan antara *public* dan *private*. Data akan terbagi 2, yaitu data yang dianggap penting akan tersimpan dalam *private cloud storage* dan data lainnya di simpan dalam *public cloud storage*. Pada *storage* ini terdapat pilihan pengaturan data mana saja yang dijadikan *private* atau publik.

Sistem penyimpanan *cloud* dapat dianggap sebagai jaringan terdistribusi pusat data yang biasanya menggunakan teknologi *cloud computing* seperti virtualisasi dan menawarkan beberapa jenis interface untuk menyimpan data.[12]

Kelebihan *cloud storage* yaitu data dapat diakses dari berbagai lokasi (misalnya dari perangkat *mobile*) tanpa perlu menyamakan perangkat keras atau perangkat lunak yang sesuai, dan sinkronisasi fitur yang memungkinkan pelanggan untuk selalu memiliki akses ke versi terbaru dari perangkat (komputer personal, laptop, *smartphone*). Penggunaan *cloud storage* dapat dimanfaatkan sebagai layanan *Copy, Backup, Synchronization of Devices*, dan *Sharing Files.*

2.5 *Protokol HTTP*

Protokol HTTP(*Hypertext Transfer Protocol*) merupakan salah satu protokol yang sering digunakan dalam pertukaran informasi. HTTP Protokol populer diaplikasikan pada aplikasi web. Protokol HTTP mempunyai dua layanan / fitur terhadap pengiriman data yaitu *request* dan *response*[13]. HTTP Protokol menggunakan URL(*Uniform Resource Locator*) sebagai alamat terhubungnya antara *client* dan *server*. Ada empat bagian dari URL yaitu:

1. *Protocol*: Protokol yang digunakan (HTTP)
2. *Hostname*: Nama domain server (sebagai contoh www.its.ac.id)
3. *Port*: *Port* yang digunakan oleh server. Jika tidak dituliskan umumnya menggunakan *port* 80
4. *Path-and-file-name*: Nama dan lokasi file yang diminta pada direktori tertentu

Pada proses *request* dan *response* data antara *client* dan *server*, protokol HTTP menyediakan beberapa *method* pengiriman diantaranya:

1. GET: *Method* ini membolehkan *client* meminta *resource* atau

pun file yang berada pada *server*.
2. HEAD: Untuk mendapatkan *header* yang diperbolehkan pada *method* GET.
3. POST: Digunakan untuk mengirim data ke *server*.
4. PUT: Meminta *server* menyimpan data.
5. DELETE: Meminta *server* menghapus data.
6. TRACE: Minta server untuk mengembalikan mendapatkan jejak dan tindakan.
7. OPSI: Digunakan untuk mengetahui daftar *method* yang didukung.
8. CONNECT: Digunakan untuk memberi tahu proxy untuk membuat koneksi ke *host* lain dan cukup membalas kontennya, tanpa mencoba mengurai atau menyimpannya. Biasanya digunakan pada koneksi SSL(*Secure Sockets Layer*) pada *proxy*.

Jika melakukan koneksi ke *server* dengan *method* GET pada sebagai contoh "*https://artificialoflaksman.com/index.php*", maka terdapat sebuah *header* yang dikirimkan sebagai contoh seperti yang ditunjukkan pada gambar 2.8.

```
Host: artificialoflaksman.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:57.0)
Gecko/20100101 Firefox/57.0
Accept: text/html,application/xhtml+xml,application/xml;
Accept-Language: id,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Cookie: _ga=GA1.2.854760115.1513554082
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

Gambar 2.8: *HTTP Header* yang dikirim

Mulanya *port* 80 pada *server* dalam keadaan *listen*, artinya *server* menunggu *client* yang terhubung dengannya. Jika *client* telah terhubung dengan *server* maka, *client* melakukan *request*. *Request* dari *client* diterima oleh *server* dengan *header* tertentu. Jika alamat atau sub direktori pada *server* valid maka, *server* akan merespon *client* dengan mengirimkan konten yang dituju. Saat proses

berlangsung, *socket* koneksi akan tetap terbuka antara *client* dengan *server*. Respon yang dikirim misalnya seperti gambar 2.9.

```
Status: HTTP/1.1 200 OK
Content-Length: 58
Content-Type: text/html
Server: Microsoft-IIS/10.0
X-Powered-By: ASP.NET
Date: Mon, 08 Jan 2018 23:42:37 GMT
```

Gambar 2.9: Respon *HTTP header* yang diterima

Untuk pemeriksaan *request client* sukses atau tidak dapat dilihat pada hasil respon dari server. Beberapa kode status HTTP yang umum dan sering ditemukan antara lain:

1. 200 - Server berhasil mengirim kembali halaman (sukses)
2. 301 - *Source* yang dituju telah dipindah ke lokasi lain
3. 400 - *Request* jelek, artinya *server* tidak mengerti dengan *request* dari *client*
4. 404 - Halaman yang diminta tidak ada
5. 500 - *Internal Server Error*. Terjadi kesalahan pada server
6. 502 - *Bad Gateway*. *Proxy* atau *Gateway* tidak dapat menerima respon dengan baik
7. 503 - Server sementara tidak tersedia

Pengiriman pada *method* GET dilakukan dengan cara membuat data dalam bentuk variabel dan nilai dari data tersebut lalu digabungkan dengan alamat URL yang dituju. Data akan diterima oleh *server* dengan cara menguraikan data yang ada pada URL tersebut berdasarkan variabel yang telah dibuat.

2.6 *Cron Jobs*

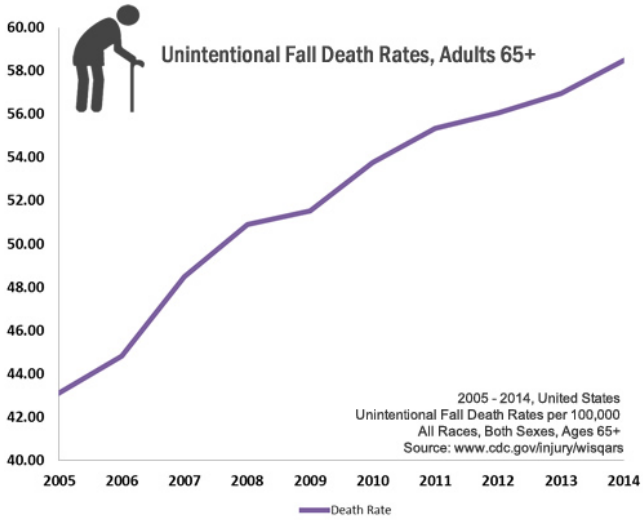
Cron jobs merupakan sebuah fitur yang dimiliki oleh *server* (*hosting*) untuk melakukan otomatisasi suatu perintah atau *script* untuk menjalankannya pada waktu tertentu seperti setiap hari, setiap minggu, jam-jam tertentu dan sebagainya. Sebagai contoh, *cron jobs* dapat memperbarui sebuah *database* untuk menghapus sebuah *user* yang sudah tidak aktif selama beberapa bulan atau dalam ku-

run waktu tertentu. Umumnya konfigurasi *cron jobs* dilakukan pada *cPanel* penyedia *hosting*, namun hal ini juga dapat dilakukan melalui terminal atau *command prompt*, dari sana semua option tentang penjadwalan bisa dilakukan. Dengan adanya fitur ini, otomatisasi tentang pembaruan data akan lebih mudah. Untuk menggunakan *cron jobs* dengan lebih kompleks, pengetahuan tentang *commands linux* sangat dibutuhkan. Ada dua tahapan dalam membuat crob job pada server.

1. Menentukan jadwal - Hal yang pertama harus dilakukan adalah menentukan jadwal kapan *script* akan dijalankan. Penentuan jadwal harus dilakukan seefektif mungkin dikarenakan penjadwalan dengan frekuensi tinggi dapat mempengaruhi kinerja server
2. Menuliskan perintah - Setelah menentukan penjadwalan selanjutnya adalah menuliskan perintah untuk menjalankan *cron jobs* sesuai jadwal yang ditentukan. *Command* terdiri dari lima tanda bintang dan diakhiri dengan perintahnya. Contoh : ******/command*. Tanda bintang tersebut merupakan penunjuk waktu yang terdiri dari 5 bagian. Dari kiri ke kanan dimulai dari menit(0-59), jam(0-23), hari dalam bulan (1-31), bulan (1-12), hari dari minggu (0-6). Sedangkan untuk command dapat diganti dengan jenis perintah yang ingin dijalankan

2.7 Resiko Jatuh pada Manula

Setiap tahun jutaan orang yang berumur 65 keatas (manula) mengalami jatuh. satu dari empat manula jatuh setiap tahunnya, tetapi kurang dari setengah yang mengatakan kepada dokter. Satu kali jatuh memiliki kemungkinan dua kali lipat untuk terjatuh di kemudian hari. Satu dari lima jatuh menyebabkan cedera serius seperti patah tulang atau cidera kepala. Lebih dari 800.000 pasien setahun dirawat di rumah sakit karena cedera jatuh, paling sering karena cedera kepala atau patah tulang pinggul. Banyak cidera atau luka yang diakibatkan oleh jatuh seperti patah tulang baik tulang tangan, pinggul ataupun pergelangan kaki. Selain patah tulang, jatuh juga dapat menyebabkan luka serius dikepala dan manula yang kepalanya berbenturan langsung saat jatuh harus segera dilaporkan



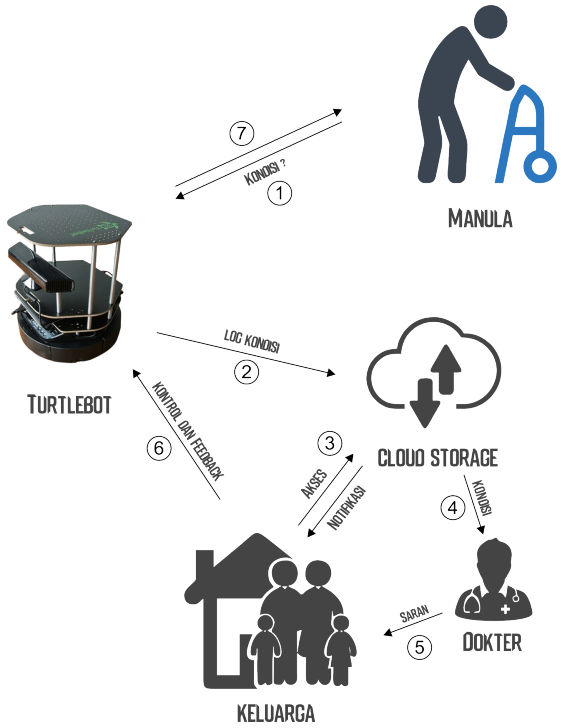
Gambar 2.10: Tingkat kematian karena jatuh yang tidak disengaja [7]

kerumah sakit untuk diperiksa apakah ada luka didalam otak atau tidak. Selain itu juga, banyak manula yang telah jatuh memiliki trauma dan rasa takut akan jatuh sehingga menyebabkan mereka takut untuk melakukan aktivitas mereka sehari-hari seperti biasanya dan menyebabkan tubuh mereka akan melemah setiap harinya karena rutinitas yang telah dihentikan.[7]

BAB 3
DESAIN DAN IMPLEMENTASI SISTEM

3.1 Desain Sistem

Tugas akhir ini bertujuan untuk melakukan pemantauan aktivitas serta kondisi manula yang tidak dapat dipantau secara langsung oleh anggota keluarga karena alasan tertentu seperti bekerja, sekolah dan lainnya sekaligus memberikan suatu notifikasi kepada kerabat terdekat jika terjadi sesuatu kepada orangtua melalui *messenger* yang dalam tugas akhir ini adalah LINE.



Gambar 3.1: Desain Sistem

Tahapan sistem pemantau kondisi manula sesuai angka pada gambar :

1. Manula mengenakan sebuah *wearable device* yang terdiri dari arduino dan sensor untuk pendeteksi jatuh atau tidaknya manula tersebut serta sensor *monitoring* detak jantung untuk mengetahui tingkat aktivitas dari manula. Tidak hanya itu, pada *wearable device* juga disediakan tombol yang berfungsi untuk memberi notifikasi pada keluarga secara langsung jika manula merasa membutuhkan bantuan. Kemudian data yang telah didapat ini akan dikirimkan ke *TurtleBot*. Demikian pula pada *TurtleBot* akan disiapkan perangkat penerima untuk menerima data yang telah dikirim tadi.

2. Akan disiapkan beberapa *discovery device* pada beberapa ruangan di dalam rumah. Saat manula dalam keadaan bahaya maka *wearable device* bisa menentukan posisinya di dalam rumah tersebut melalui *cubeacon* pada *wearable device* yang akan terdeteksi oleh *discovery device*, kemudian saat sinyal tanda bahaya dikirimkan TurtleBot akan datang ketempat sinyal itu berada dan akan mengambil gambar sekitar ruangan yang akan diteruskan ke notifikasi.

3. *TurtleBot* yang memiliki *access point* akan mengirimkan data yang telah diterima menuju *cloud storage*. *Log* kondisi manula dalam waktu tertentu akan disimpan di *database* yang berbeda dengan *database* utama. Tujuannya agar dapat diakses oleh keluarga manula ataupun dokter secara *online*.

4. Jika manula dalam keadaan bahaya seperti terjatuh atau melakukan aktivitas yang berat maka otomatis *wearable device* akan ter-*trigger* untuk mengirim notifikasi melalui LINE baik secara individu (per orang) ataupun secara kelompok (*group*). Selain itu keluarga juga dapat memantau secara langsung kondisi terkini manula melalui sebuah *web application* .

5. Selain dikirim ke keluarga manula, pesan bahaya juga dikirim ke dokter pribadi yang menangani manula tersebut. Log mengenai kondisi juga bisa diakses oleh dokter tersebut, tujuannya adalah agar dokter dapat melakukan analisa terhadap

kondisi manula baik dari aktivitas ataupun keadaan kesehatan manula tersebut. Dokter yang telah melakukan analisa dapat memberikan saran kepada keluarga untuk penanganan sehari-hari terhadap manula melalui *web application* yang digunakan untuk memantau tadi.

6. Keluarga yang memantau dapat memberikan saran kepada manula ataupun memberikan perhatian secara langsung melalui *TurtleBot* dan *messenger*. Anggota keluarga yang menuliskan *keyword* tertentu pada *messenger* dapat men-*trigger* *TurtleBot* untuk melakukan sesuatu dan menampilkan pesan yang ingin disampaikan pada manula melalui *LCD Screen* yang telah disiapkan pada *TurtleBot*.

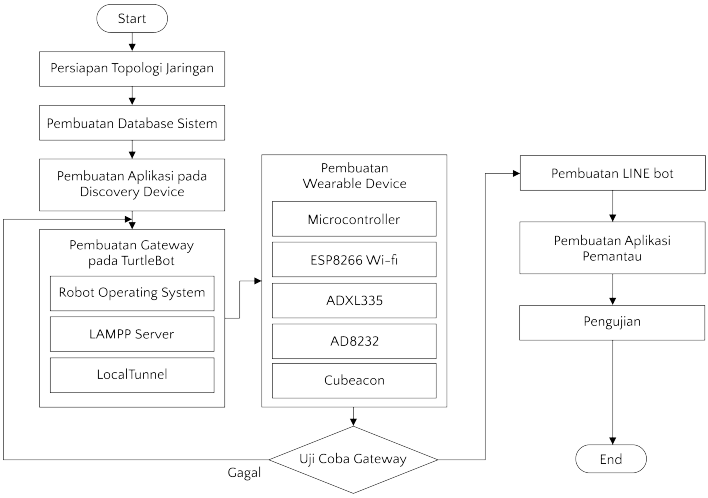
7. Pesan yang dikirim oleh anggota keluarga tadi ditampilkan oleh *TurtleBot* untuk ditunjukkan kepada manula sehingga manula tidak merasa kesepian dan selalu diperhatikan oleh anggota keluarganya.

3.2 Alur Implementasi Sistem

Secara rinci Alur Implementasi dalam pengerjaan tugas akhir ini terbagi menjadi tujuh tahapan proses dan satu percabangan seperti yang terlihat pada Gambar 3.2.

3.3 Desain Jaringan

Desain jaringan pada penelitian ini melibatkan beberapa komponen. Pada *wearable device*, data akan dikirim melalui *port* 80 untuk mengirimkan data dari sensor menuju *gateway*. *Gateway* dalam penelitian ini adalah *TurtleBot* yang dihubungkan dengan sebuah *netbook* dan *repeater* untuk mengakses internet. *Notebook* akan membuat sebuah *secure tunnel* menuju internet dengan me-*forward* *port* sehingga *localhost* pada *notebook* dapat diakses oleh seluruh pengguna internet, namun tidak semua *port* yang dibuka melainkan hanya *port* 22 saja karena *port* 22 merupakan protokol untuk SSH, sehingga saat melakukan koneksi, pengguna internet dan *gateway* harus sama-sama memiliki *key* agar dapat saling berkoneksi. De-



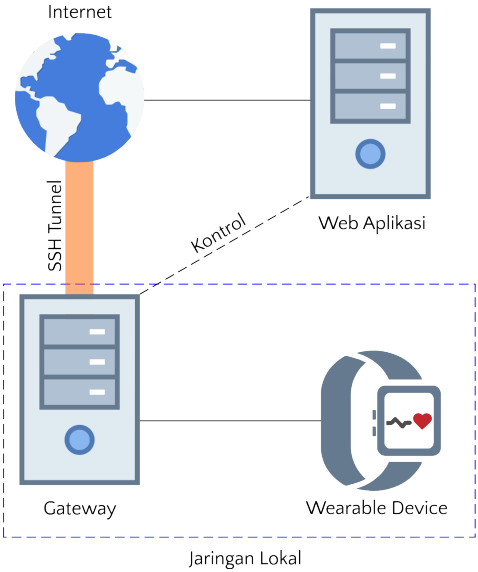
Gambar 3.2: Alur Implementasi Sistem

ngan adanya *tunnel*, web aplikasi dapat melakukan kontrol terhadap *TurtleBot*.

3.4 Desain *Database*

Database yang digunakan dalam tugas akhir ini memerlukan 9 tabel agar sistem berjalan dengan baik seperti pada gambar 3.4.

1. Tabel pertama adalah *tb_profileManula*, tabel ini adalah tabel yang digunakan untuk menghimpun seluruh nama dan usia manula yang terdaftar pada sistem serta *wearable device id* yang dikenakan pada masing-masing manula.
2. Tabel kedua adalah *tb_discoveryDevice*, tabel ini adalah tabel yang digunakan untuk menghimpun *receiver id* dan lokasi yang ditunjuk pada *receiver device*. Tabel ini akan menjadi penentu suatu *device* berada dimana dalam suatu lokasi, sehingga *device* dapat di *assign* pada lokasi yang berbeda.
3. Tabel ketiga adalah *tb_location*, tabel ini adalah tabel penentu lokasi manula yang menggunakan *wearable device*. Tabel ini diturunkan dari dua tabel sebelumnya sebagai *foreign key* dan kolom yang diambil adalah *wearable device id* yang mana pada



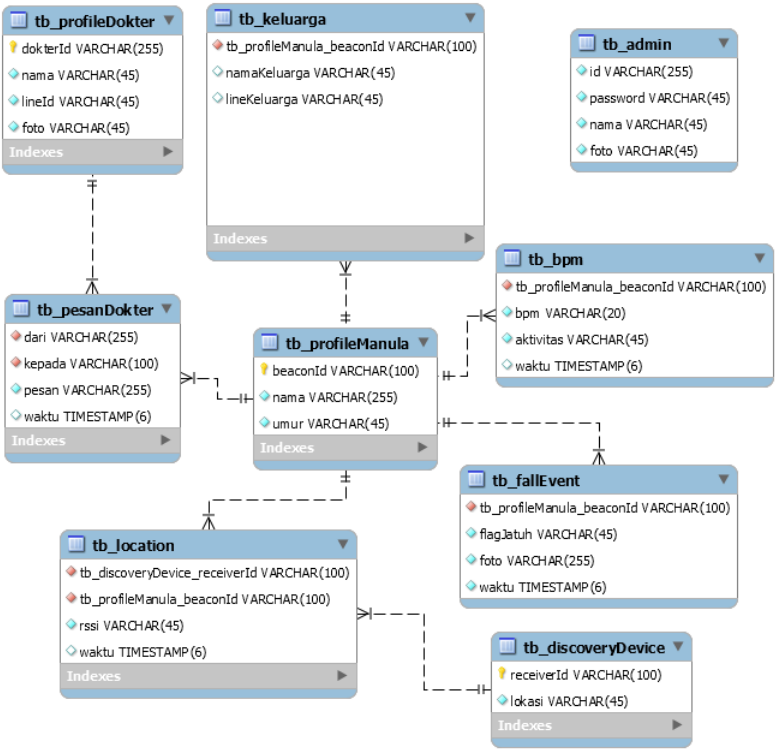
Gambar 3.3: Desain Jaringan

gambar ditunjukkan sebagai *beaconId* serta kolom *device id* yang mana pada gambar ditunjukkan sebagai *receiverId*.

4. Tabel keempat adalah *tb_bpm*, tabel ini adalah tabel yang digunakan sebagai *log* untuk menyimpan *beats per minute* pada manula yang mengenakan *wearable device*. pada tabel ini juga diturunkan beberapa kolom dari tabel sebelumnya sebagai *foreign key* yaitu kolom *beaconId*.
5. Tabel kelima adalah *tb_fallEvent*, tabel ini berisi *log* jika manula yang menggunakan *wearable device* terdeteksi jatuh. Sama halnya dengan *tb_bpm*, tabel ini juga diturunkan kolom yang sama dari *tb_profileManula* sebagai *foreign key*.
6. Tabel keenam adalah *tb_keluarga*, tabel ini berisi keluarga dari manula yang dipantau. Tujuan dari adanya tabel ini adalah untuk mengirim notifikasi kepada keluarga sesuai dengan manula yang mengenakan *device*.
7. tabel ketujuh adalah *tb_profileDokter*, tabel ini berisi profil dokter yang menangani manula sehingga selain mengirim no-

tifikasi ke keluarga, notifikasi juga akan dikirim ke dokter.

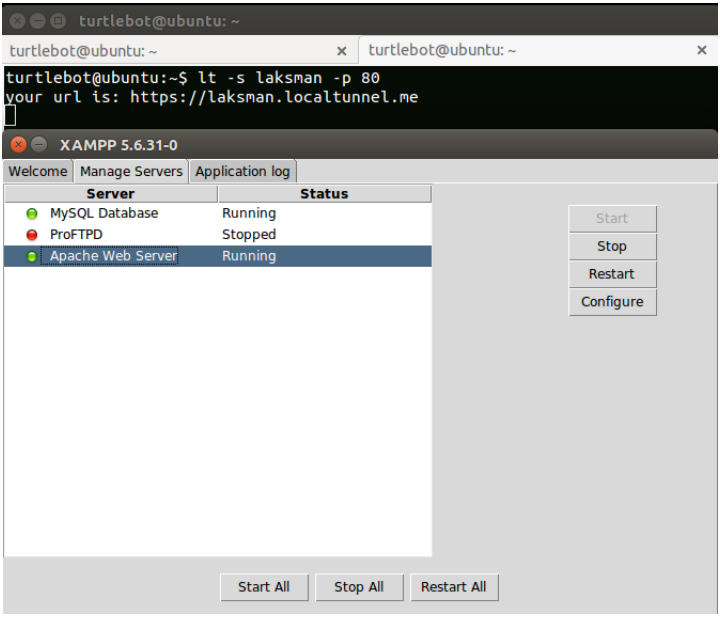
8. tabel kedelapan adalah *tb_pesanDokter*, tabel ini berisi segala pesan yang diberikan dari dokter untuk manula, pesan ini berisi saran yang nantinya dapat dibaca oleh keluarga.
9. tabel kesembilan adalah *tb_admin*, tabel ini berisi daftar para *admin* berupa *username* dan *password* untuk memantau sistem melalui aplikasi *admin*.



Gambar 3.4: Desain Database

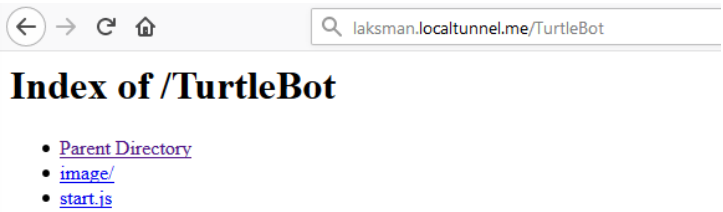
3.5 Pembuatan *Gateway* Pada TurtleBot

Gateway pada TurtleBot digunakan sebagai media penyimpanan dan media komunikasi data yang diterima dari *wearable device*. Segala data yang dikirim oleh *wearable device* akan dikirim melalui protokol *wi-fi* menuju *gateway* dalam jaringan lokal. Satu *gateway* pada TurtleBot terdiri atas satu buah *notebook* atau *single board computer* yang memiliki sistem operasi Ubuntu serta memiliki *Robot Operating System* sebagai *middleware*. Gateway dalam kasus ini didesain sebagai penghubung jaringan lokal dan jaringan internet, segala data yang dikirim dari manula menuju *cloud storage* harus melalui *gateway* dahulu dengan tujuan data yang dikirim dari manula tersimpan dahulu pada *gateway* baru kemudian melewati jaringan internet menuju *cloud storage*. Konsep ini bertujuan agar data historis pada manula bersifat *redundant*, maksud dari *redundant* adalah data yang sama tersimpan di *local gateway* dan *cloud storage*.



Gambar 3.5: Memulai *Web-Server* dan membuat *Tunnel*

TurtleBot dalam hal ini tidak hanya menjadi *gateway* saja namun juga akan menjalankan fungsinya sebagai robot yaitu mengambil gambar manula yang terjatuh, maka dari itu *gateway* pada TurtleBot akan dibuatkan sebuah *tunnel* dengan protokol *SSH* dengan tujuan agar gambar yang tersimpan di lokal bisa dibaca oleh *back-end process* yang ada pada *cloud server*. Tujuan dari penggunaan protokol *SSH* adalah agar komunikasi yang terjadi antara *cloud server* dan *local server* bersifat *secure*. Untuk membuat TurtleBot menjadi sebuah *gateway*, hal yang harus dilakukan adalah memasang sebuah *web-server software*, *database management system* dan konfigurasi *tunnel*. *Web-server software* yang digunakan untuk membuat *gateway* pada tugas akhir ini adalah *apache* dan untuk *database management system* yang digunakan adalah *MySQL*. Pada tugas akhir ini kedua hal tersebut dipasang dalam satu *bundle* yaitu *XAMPP*. Setelah konfigurasi *web-server* dan *database* melalui *XAMPP*, tahap selanjutnya adalah melakukan konfigurasi *tunnel*. Konfigurasi dilakukan dengan menuliskan sebuah *script* berbasis node.js, dengan cara seperti ini, robot dapat mengekspos *local storage* ke jaringan internet dengan *domain name server* yang sudah dikonfigurasi pada node.js. *Script tunnel* hanya bisa dijalankan jika *web-server* pada *gateway* sudah dinyalakan.



Gambar 3.6: Akses *tunnel* melalui internet

Seperti yang ditunjukkan oleh gambar 3.6, ketika *web-server* dan *tunnel* telah dinyalakan maka TurtleBot saat ini sudah berfungsi menjadi *gateway* karena dapat mengekspos file lokal ke jaring-

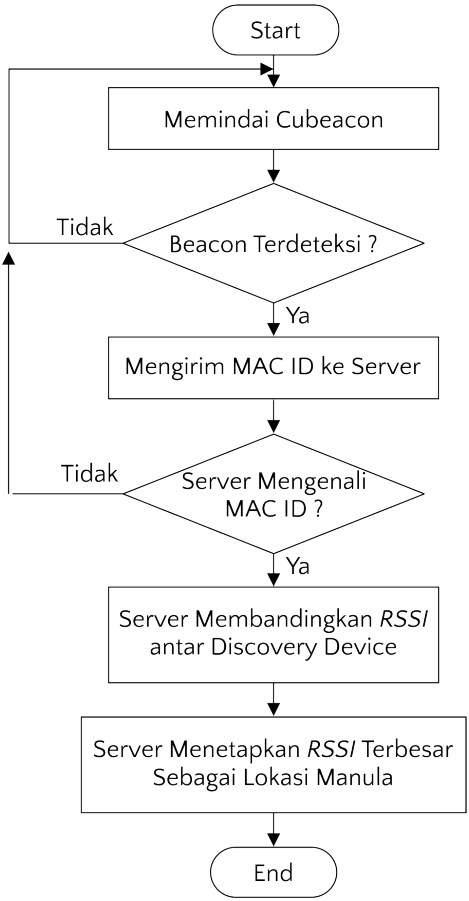
an internet sehingga dalam kondisi ini, TurtleBot dapat melakukan *push image* ke *database* yang ada pada *cloud server* serta dapat meneruskan data yang diterima dari *wearable device* menuju *cloud storage*

3.6 Pembuatan *Discovery Device* Berbasis Aplikasi Android

Aplikasi *discovery* adalah aplikasi yang digunakan untuk mendeteksi lokasi dari manula yang mengenakan *wearable device*, aplikasi *discovery* ini merupakan aplikasi berbasis android. *Smartphone* android yang telah terpasang aplikasi ini akan diletakkan di titik tertentu didalam rumah sehingga dapat melakukan deteksi terhadap keberadaan *wearable device*.

Discovery device akan melakukan deteksi secara terus menerus dalam rentang waktu 10 detik sambil mengirim *MAC ID* yang terpindai ke *server*. *Server* akan menentukan apakah *MAC ID* yang terpindai adalah *MAC ID* yang terdaftar pada database sebagai *MAC ID wearable device*. Jika iya maka server akan membandingkan kekuatan sinyal yang diterima (*RSSI*) pada *discovery device* satu dengan yang lainnya, *discovery device* yang menerima kekuatan sinyal paling besar atau terdekat terhadap *wearable device*, maka lokasi *discovery device* itu akan dijadikan patokan lokasi manula yang mengenakan *wearable device*.

RSSI, *ID discovery device* dan *MAC ID* dari *cubeacon* akan disimpan dalam *database* melalui proses *update*, setiap kali proses *update* terjadi maka waktu tersimpannya data tersebut juga akan ikut ter-*update*. Hal ini bertujuan agar keluarga yang memantau dapat mengetahui interval waktu keberadaan manula dalam suatu lokasi. Sebagai contoh, manula yang terpindai di ruang kamar pada pukul 9 kemudian tidak ada perubahan sampai pukul 10, maka anggota keluarga dapat mengasumsikan bahwa manula sedang beristirahat. Secara rinci, cara kerja *discovery device* digambarkan seperti diagram pada gambar 3.7



Gambar 3.7: Proses pemindaian *wearable device*

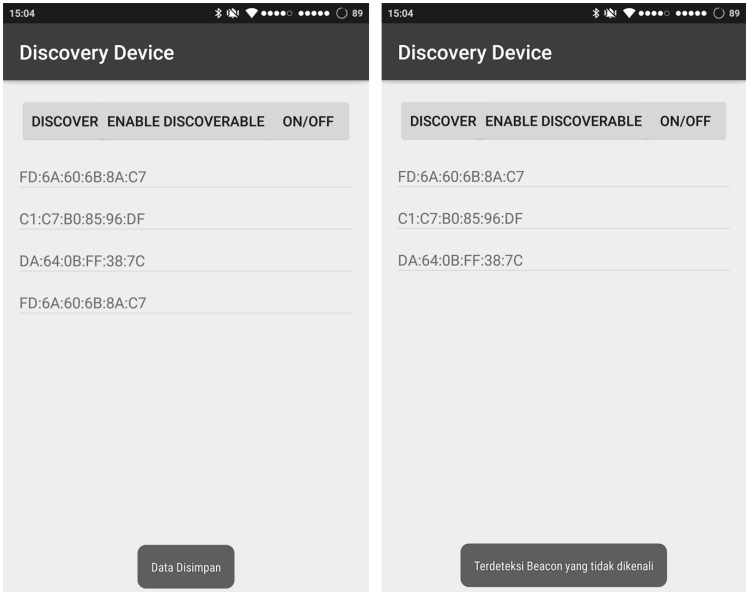
3.6.1 *User Interface* Aplikasi

Aplikasi yang digunakan pada *discovery device* dikembangkan dengan menggunakan Android Studio versi 2.3.3. Aplikasi ini hanya memiliki 2 *user interface*, yaitu *UI* saat melakukan *assign* lokasi terhadap *discovery device* dan *UI* saat melakukan pemindaian cubeacon. *Discovery device* yang dalam hal ini adalah *smartphone*

dapat dipindah-pindah sesuai kebutuhan untuk mengetahui lokasi dari manula. Pada *UI assign* lokasi, *user* hanya disediakan beberapa tombol yang dimaksudkan untuk melakukan *assign smartphone* tersebut pada lokasi yang sudah ditentukan pada *database*. Sebagai contoh, pada *database* telah di *assign* bahwa tombol lokasi 1 adalah ruang kamar, maka saat *smartphone* di *assign* sebagai lokasi 1 oleh *user* dan saat manula berada dekat dengan *smartphone* itu, manula akan dianggap berada pada ruang kamar pada *server*. Proses *assign* tombol terhadap lokasi dapat dilakukan pada *dashboard admin* yang akan dijelaskan kemudian. *UI assign* lokasi terhadap *smartphone* dapat dilihat pada gambar 3.8

Gambar 3.8: User Interface *Assign* Lokasi

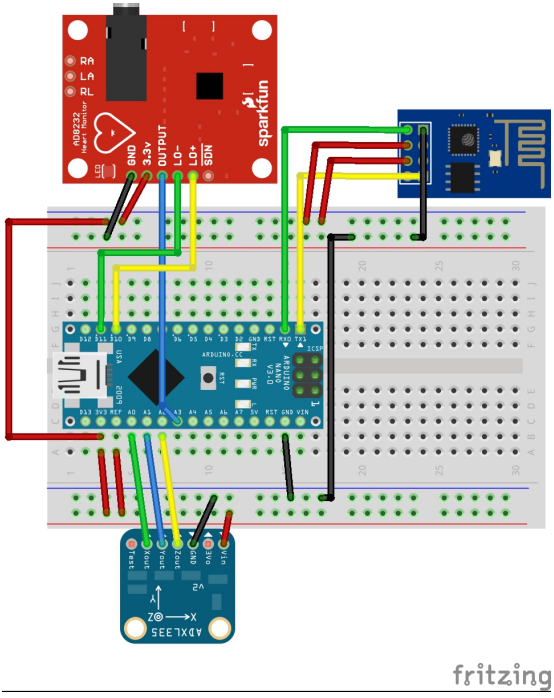
Pada *user interface* pemindai cubeacon, *activity* pada aplikasi dirancang agar semua *MAC ID* dari *device* yang memancarkan sinyal *bluetooth v4.0* terpindai oleh aplikasi setiap 10 detik sekali, dalam 10 detik itu aplikasi diberi kesempatan untuk mengirim *RSSI* yang diterima beserta *MAC ID* yang terpindai menuju *server* untuk diproses pada *back-end process*. Setelah itu *server* akan memberi jawaban kembali kepada aplikasi, jika *MAC* yang dipindai sudah terdaftar pada database maka aplikasi akan menerima jawaban dan menampilkan sebuah *pop-up* "Data tersimpan" pada *user interface* aplikasi. Namun jika server tidak mengenali *MAC ID* yang terpindai atau dengan kata lain *MAC ID* tersebut tidak terdaftar pada *database*, aplikasi akan menampilkan sebuah *pop-up* "Terdeteksi *beacon* yang tidak dikenali" pada *user interface* seperti yang ditunjukkan pada gambar 3.9.



Gambar 3.9: User Interface Pemindaian Cubeacon

3.7 Pembuatan *Wearable Device*

Wearable device adalah sebuah *device* yang dikenakan manula untuk mengawasi atau memantau manula dari beberapa kondisi. Kondisi yang dapat dipantau oleh *device* ini adalah kondisi *vital sign* yaitu detak jantung per menit manula, lokasi manula didalam suatu rumah, dan status jatuh atau tidaknya manula. Device ini memiliki beberapa sensor yang dapat mengambil data berkaitan dengan hal-hal yang dapat dipantau diatas. *Device* mengirim dan menerima data melalui protokol *wi-fi* dan *bluetooth low energy*. Secara detail, *wearable device* memiliki skema seperti yang ditunjukkan pada gambar 3.10. *Device* ini dikenakan pada pinggang manula dengan bantuan sabuk karena pada posisi ini modul dapat mendeteksi kondisi jatuh dengan sangat baik [14].



Gambar 3.10: Skematik *wearable device*

3.7.1 Akuisisi Data Sensor Pendeteksi Jatuh

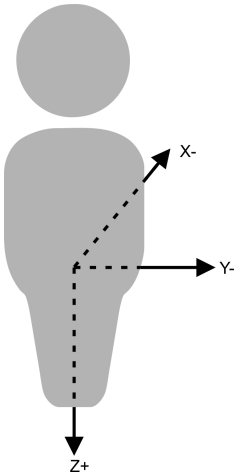
Sensor yang digunakan untuk mendeteksi jatuh pada *device* ini menggunakan modul *accelerometer*, modul *accelerometer* sudah sangat cukup untuk mendeteksi kondisi jatuh dari seseorang tanpa memerlukan bantuan sensor lainnya [15]. Metode yang digunakan untuk mendeteksi kondisi jatuh dari manula adalah *Sum Vector Magnitude (SVM)* yaitu menjumlahkan kuadrat dari percepatan pada masing-masing sumbu kemudian mengakarkannya [16]. Percepatan ini nantinya akan di learning untuk mendapatkan *threshold* yang tepat dan akan digunakan sebagai acuan untuk mendeteksi jatuh.

$$A_{SVM} = \sqrt{A_x^2 + A_y^2 + A_z^2} \tag{3.1}$$

Selain percepatan, sudut juga merupakan komponen penting yang harus diperhatikan saat mendeteksi jatuh atau tidaknya manula. Percepatan bisa saja berubah dan memenuhi *threshold* terjatuh pada saat manula berjalan biasa atau mempercepat kecepatan jalannya, maka dari itu faktor sudut juga harus diperhitungkan agar pendeteksian jatuh semakin *valid*. Dalam tugas akhir ini, sudut dapat dihitung dari percepatan yang didapat dari setiap sumbu dengan asumsi sudut yang dihitung adalah sudut terhadap sumbu x (arah depan dan belakang) dari *device*.

$$\theta = tan^{-1}\left(\frac{\sqrt{A_y^2 + A_z^2}}{A_x}\right) \times \frac{180}{\pi} \tag{3.2}$$

Mulanya *device* akan melakukan pengecekan terhadap percepatan yang terjadi pada modul *accelerometer*, jika modul memenuhi *threshold* percepatan sesuai metode yang digunakan, maka *device* akan melakukan pengecekan terhadap *angle* pada sumbu x seperti yang ditunjukkan pada gambar 3.11. Jika sudut memenuhi sudut yang ditentukan maka *device* akan menganggap manula terjatuh. Awalnya, data yang ditangkap oleh modul *accelerometer* adalah data *raw*, data *raw* ini kemudian diubah kedalam satuan *g*. Data me-

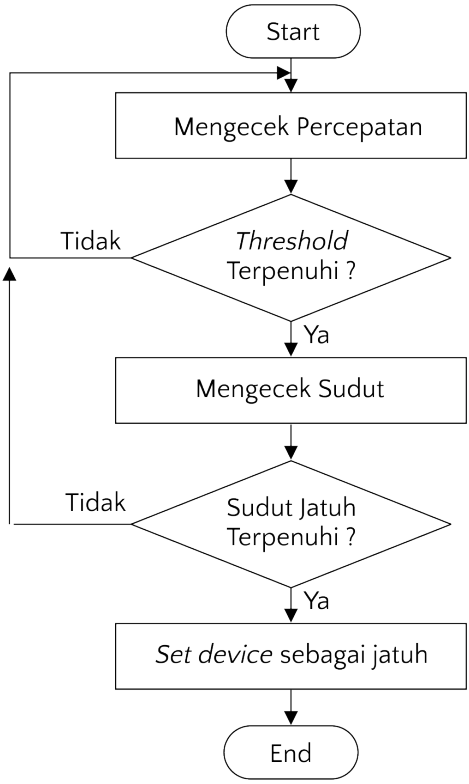


Gambar 3.11: Sumbu pada modul *accelerometer* terhadap manula

miliki rentang nilai antara 0 sampai 1 g, data g ini kemudian akan dikonversi dalam satuan percepatan (m/s^2) dengan acuan bahwa $1g = 9,8m/s^2$. Data yang sudah diolah menjadi percepatan inilah yang akan digunakan sebagai *threshold* dalam penentu jatuh atau tidaknya seorang manula. Secara rinci cara *device* melakukan pengecekan terhadap jatuh atau tidaknya manula dapat dilihat seperti gambar 3.12.

3.7.2 Akuisisi Data Sensor Denyut Jantung

Sensor yang digunakan untuk menghitung *beats per minute (BPM)* dari manula adalah modul *ECG AD-8232*. Modul ini menggunakan konektor *3.5mm* untuk mendeteksi detak jantung dari manula. Konektor memiliki 3 kutub yang akan ditempelkan pada badan manula. Karena menggunakan 5v, maka program akan melihat *peak* (puncak) dari gelombang dengan interval 0-5000 mv yang dihasilkan dari data sebagai satu denyut jantung. Setelah itu tentukan *threshold* yang tepat untuk penentuan satu denyut jantung dari gelombang yang dihasilkan. Untuk menentukan denyut jantung per menit, maka program harus menghitung jumlah *peak* yang terjadi



Gambar 3.12: Proses Penentuan Kondisi Jatuh

dalam rentang satu menit. Dalam tugas akhir ini penulis mengambil rentang waktu satu menit karena dalam rentang waktu itu *BPM* yang didapat lebih akurat daripada menggunakan rentang waktu 15 detik atau 30 detik.

3.7.3 Proses Pengiriman Data

Proses pengiriman data dilakukan oleh modul *wi-fi* pada *wearable device* menuju robot secara singkat dapat dilihat pada kotak *Algorithm* 1. Secara detil, proses yang terjadi pertama kali adalah melakukan koneksi dengan *router* atau modem dengan *command*

Algorithm 1: Proses Pengiriman Data

1 $AT+CWJAP$ = "nama$SSID$","$passwordSSID$"
2 $AT+CIPSTART$ = "TCP/UDP","$host$",$port$
3 $AT+CIPSEND$ = jumlah$Character$

AT+CWJAP = "namaSSID","passwordSSID".

Setelah melakukan koneksi dengan *router* atau modem, modul yang dalam hal ini sudah tergabung dalam *wearable device* akan mendapatkan *IP*. Setelah mendapat *IP*, modul akan mencoba melakukan *handshaking* dengan *server* yang dalam hal ini adalah *robot*. *handshaking* dilakukan dengan menggunakan *command AT+CIPSTART = "TCP/UDP","host",port*. Jika proses *handshaking* berhasil maka modul akan mendapat balasan dari *server* dan akan memberi respon *"CONNECT OK"*, tetapi jika gagal, modul akan memberi respon *"CLOSED"*.

Setelah berhasil melakukan *handshaking*, modul telah siap untuk mengirim data menuju robot, proses pengiriman data menggunakan *command AT+CIPSEND = jumlahCharacter*. Setelah mengirim *command* tersebut, modul akan meminta *url back end* dimana data akan diproses, sebagai contoh *GET /tugasAkhir/olahdata.php?bpm=varBPM*. Jika proses berhasil, modul akan memberi respon *200 OK* yang berarti data telah diterima oleh *server*.

3.7.4 Desain *Wearable Device*

Wearable device memiliki dimensi 9cm x 6cm x 4cm, memiliki 1 *switch on/off*, 1 konektor 3,5mm, dan 1 port *mini usb* untuk konfigurasi. *Device* ini terbuat dari akrilik dengan ketebalan 2mm berwarna hitam serta memiliki 6 lubang bergaris yang berguna sebagai ventilasi agar *device* tidak panas dan transmisi data berjalan dengan lancar. Untuk lebih detailnya, *wearable device* terlihat seperti gambar 3.13.

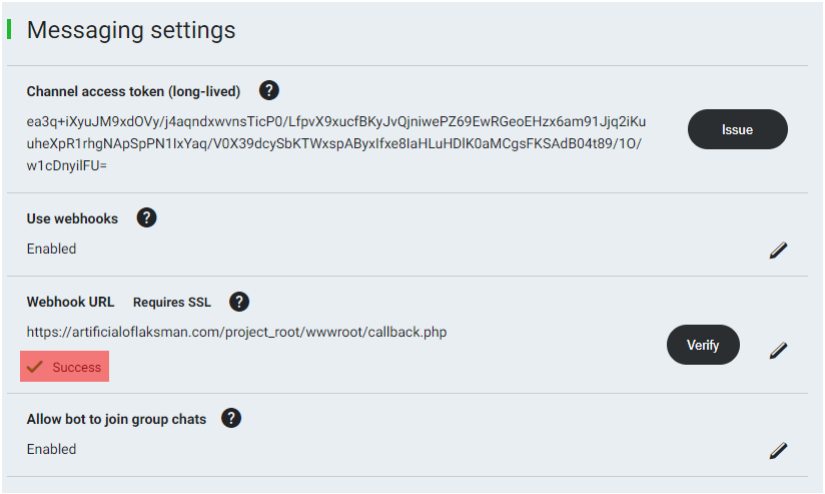
Gambar 3.13: Desain *Wearable Device*

3.8 Desain Sistem Notifikasi Menggunakan LINE BOT

LINE BOT dalam tugas akhir ini digunakan sebagai pemantau dan notifikasi kepada kerabat terdekat. Pada tugas akhir ini, bot dikembangkan dengan menggunakan bahasa pemrograman PHP dan berjalan pada *server Microsoft* dengan *SSL Certificate*. *Bot* tidak akan berjalan jika *server* tidak tersertifikasi SSL (*Secure Socket Layer*). Dalam pengembangan *bot* ini, penulis menggunakan metode webhook untuk melayani user yang menggunakan *bot* ini, sehingga setiap *user* yang menggunakan *bot* ini akan dilayani dengan *thread* masing-masing. Setiap bot yang telah dibuat dapat dilakukan pengaturan melalui *developer console* LINE.

Hal terpenting adalah mengatur *setting webhook* dengan menggunakan *url* yang telah diprogram sebelumnya pada *console*. Pada tugas akhir ini, penulis menaruh *script* php yang menjadi pusat pelayanan para *user* saat memantau kondisi manula pada *url https://artificialoflaksman.com/project_root/wwwroot/callback.php*. *Script* yang berjalan lancar akan memiliki tanda *succes* setelah melakukan *verify* yang terlihat seperti gambar 3.14.

Ada beberapa layanan yang disediakan oleh bot pada tugas

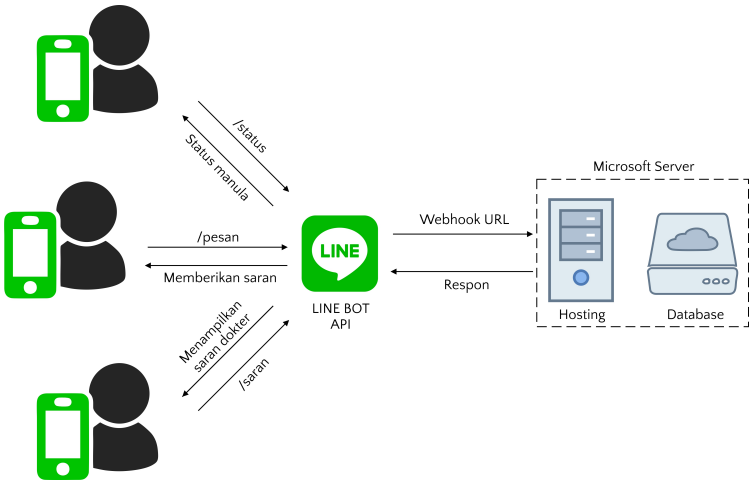


Gambar 3.14: Konfigurasi *Webhook*

akhir ini seperti yang ditunjukkan pada gambar 3.15. Sebagai contoh, saat keluarga mencoba memantau kondisi manula berdasarkan *id device*. Setiap orang yang menggunakan *wearable device* seperti yang digambarkan pada *subsection* sebelumnya, mereka dapat menjadikan bot ini sebagai teman dan kemudian menggunakan *command* /status disertai dengan *device id*. Sebagai contoh, *device id* adalah 531853, *user* yang ingin memantau manula yang mneggunakan *wearable device* tersebut hanya perlu menuliskan /status 531853 pada chat untuk mendapat status dari manula seperti yang terlihat pada gambar 3.16.

Saat *user* menulis status disertai *device id*, *bot* akan membalas dengan memberikan *interface carousel* dengan beberapa informasi didalamnya, informasi yang diberikan oleh *bot* meliputi foto manula yang mengenakan *device*, nama manula, *bpm* yang terakhir kali diambil beserta waktu pengambilan terakhir, tingkat aktivitas yang dikonversi dari *bpm* serta lokasi. Selain itu juga ada dua informasi tambahan yaitu *link* untuk menuju ke aplikasi pemantau yang akan dijelaskan kemudian, serta saran dari dokter.

Untuk sistem notifikasi jatuh, *bot* akan melakukan *push messa-*

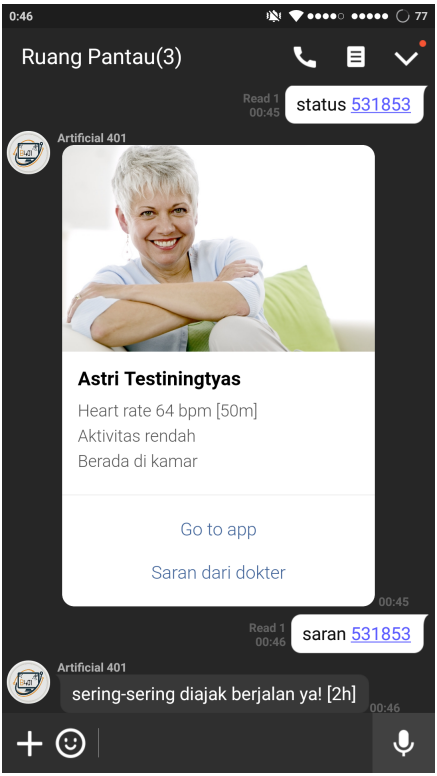


Gambar 3.15: Alur komunikasi LINE *bot*

ge kepada *user* yang *id* nya terdaftar pada *database* sebagai keluarga dari manula, bot akan otomatis melakukan *push message* ketika *device* memenuhi *threshold* terjatuh yang telah dijelaskan pada *subsection* sebelumnya. Notifikasi yang dikirim oleh bot adalah berupa pesan dan berupa gambar yang diambil oleh *TurtleBot* saat jatuh terjadi. Notifikasi yang diterima ditunjukkan seperti pada gambar 3.17.

3.9 Aplikasi Pemantau

Aplikasi pemantau merupakan aplikasi untuk memantau kondisi serta aktivitas manula. Aplikasi ini dikembangkan untuk *multiplatform* sehingga anggota keluarga dapat memantau dengan lebih leluasa. Aplikasi ini dikembangkan dengan menggunakan *materialize css* dengan sifat *responsive* sehingga dapat digunakan pada segala jenis *device*. Aplikasi pemantau ini memiliki beberapa komponen didalamnya diantaranya adalah *profile*, *status* dan *log BPM*. *profile* berisi foto yang dapat di *pop up*, nama, dan umur dari manula yang mengenakan *device*, status berisi beberapa kondisi yang dapat dipantau oleh *device* seperti lokasi, denyut jantung, tingkat



Gambar 3.16: LINE bot untuk memantau kondisi

aktivitas dan status jatuh, kemudian *log BPM* berisi grafik dari denyut jantung manula dalam satu hari dan akan diganti per hari. Pergantian dari grafik dilakukan oleh *cron jobs* yang diatur pada *server*. Secara rinci, *user interface* aplikasi pemantau ditunjukkan seperti gambar 3.18.

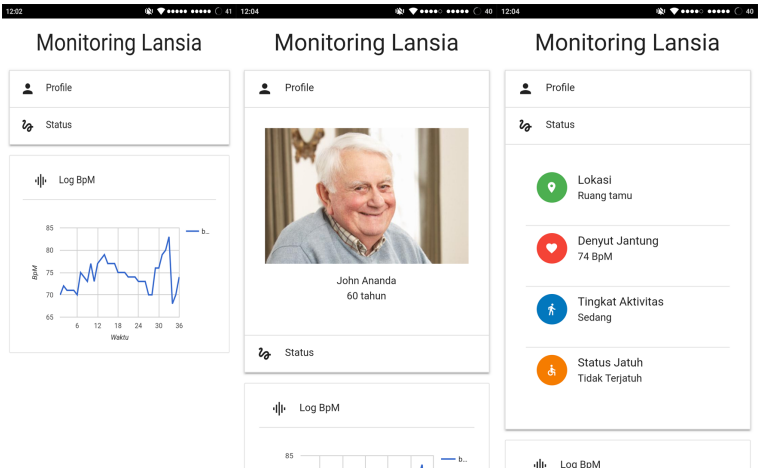
3.10 *Admin Dashboard*

Aplikasi *Admin Dashboard* merupakan aplikasi yang digunakan untuk melakukan penetapan *device* yang dikenakan oleh manula dengan identitas manula dan dengan keluarga manula. Aplikasi di-

Gambar 3.17: Notifikasi Jatuh

kembangkan secara *hybrid* dimana aplikasi ini dapat menjadi aplikasi *native* dan dapat pula dibuka dengan menggunakan *browser*. Aplikasi dapat menjadi *native* dengan syarat pengguna aplikasi memiliki *browser* Google Chrome didalam perangkat *mobile* pengguna. Aplikasi *dashboard* dikembangkan dengan menggunakan *framework bootstrap*.

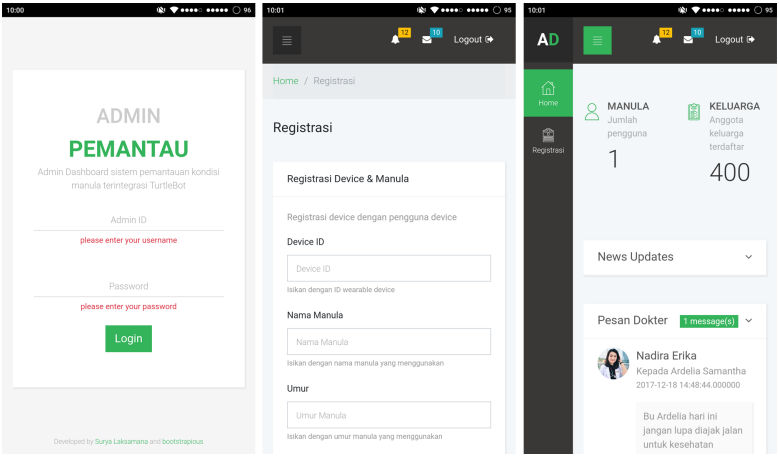
Aplikasi ini memiliki beberapa komponen didalamnya. Pertama adalah *form login*, *form* ini digunakan untuk memasukkan *credentials* dari admin yang akan melakukan penetapan *device* dengan identitas manula dan keluarga manula. *Credentials* ini akan digunakan menjadi sebuah *session* dalam PHP untuk menampilkan



Gambar 3.18: *User Interface* Aplikasi Pemantau Berbasis *Web*

identitas dan foto *admin*.

Komponen kedua adalah home, pada *home* ini semua informasi dasar atau alur komunikasi yang terjadi antara dokter, manula



Gambar 3.19: *User Interface* Aplikasi *Dashboard Admin*

dan keluarga ditampilkan. Segala pesan yang dikirim oleh dokter melalui LINE *bot* akan tampil pada halaman ini untuk dipantau. Halaman *home* juga akan menampilkan berapa *device* yang sudah terinstal pada sistem dan berapa keluarga yang sudah terdaftar dengan *device*.

Komponen ketiga adalah registrasi, pada halaman ini *admin* dapat melakukan penetapan *device* dengan identitas manula, penetapan manula dengan keluarga manula untuk mendapat notifikasi. Tampilan aplikasi *dashboard* dapat dilihat pada gambar 3.19.

BAB 4
PENGUJIAN DAN ANALISA

Pada bab ini, dilakukan pengujian terhadap sistem yang telah dibuat. Pengujian dilakukan guna mengetahui tingkat akurasi dan tingkat efisiensi sistem serta menarik kesimpulan dari sistem yang telah dibuat. Secara garis besar, pengujian yang akan dilakukan dalam pengujian ini antara lain:

1. Pengujian *Wearable Device* - meliputi pengujian untuk tingkat akurasi sensor baik sensor pendeteksi jatuh ataupun sensor pendeteksi denyut jantung
2. Pengujian *Cloud Server* - meliputi waktu respon dan keandalan *server*
3. *Pengujian Gateway* - meliputi waktu respon, *path planning* dan akurasi cubeacon mendeteksi lokasi untuk *path planning*
4. Pengujian Aplikasi Pemantau - meliputi *User Experience* dan efektifitas *User Interface* serta pengujian waktu yang digunakan untuk melakukan *load data*

4.1 Pengujian *Wearable Device*

Sebelum melakukan pengujian pada *wearable device*, sukarelawan harus mengenakan *wearable device* dengan dimensi 9 x 6 x 4 cm dan perkiraan berat 185 gram pada bagian pinggang dengan bantuan *belt* atau sejenisnya. Penggunaan *wearable device* pada bagian pinggang adalah penggunaan pada posisi paling baik untuk mendeteksi adanya kondisi jatuh dari sukarelawan. Kemudian, penggunaan konektor untuk mendeteksi *beats per minute* dari sukarelawan juga disesuaikan dengan dokumentasi yang ada. Pemasangan *wearable device* pada sukarelawan terlihat seperti pada gambar 4.1. Pengujian pada bagian ini meliputi tingkat akurasi dari modul *accelerometer* untuk mendeteksi jatuh, tingkat keberhasilan untuk mendeteksi denyut jantung serta tingkat keberhasilan pengiriman data dari *wearable device* menuju *gateway*

Gambar 4.1: Pemasangan *wearable device* pada sukarelawan

4.1.1 Akurasi Modul *Accelerometer* Sebagai Pendeteksi Kondisi Jatuh

Pada poin ini, akan diuji seberapa baik tingkat akurasi yang dimiliki *3-axis accelerometer* dalam mendeteksi adanya kondisi jatuh pada beberapa kondisi. Pengujian dilakukan sebanyak 10 kali dalam setiap kondisi, kemudian dihitung berapa akurasi dari sensor *accelerometer* dalam pendeteksian jatuh. Hasil pengujian *true positive* ditunjukkan pada tabel 4.1 sedangkan hasil pengujian *false negative* ditunjukkan pada tabel 4.2.

Dari pengujian dengan kondisi *true positive* yang dilakukan, modul *accelerometer* menunjukkan hasil yang baik, dari tiga kondisi yang diberikan, modul *accelerometer* dapat mendeteksi jatuh dengan akurasi 96,67%. Kondisi yang pertama adalah kondisi di-

Tabel 4.1: Tabel pengujian modul accelerometer kondisi true positive

Jenis Pengujian	Jumlah Pengujian	Terdeteksi	Tidak Terdeteksi	Akurasi
Terjatuh bebas ke depan	10	10	0	100%
Terjatuh bebas ke samping	10	9	1	90%
Terjatuh bebas ke belakang	10	10	0	100%
Rata-rata akurasi				96.67%

Tabel 4.2: Tabel pengujian modul *accelerometer* kondisi *false positive*

Jenis Pengujian	Jumlah Pengujian	Terdeteksi	Tidak Terdeteksi	Akurasi
Berdiri kemudian duduk	10	8	2	20%
Lompat kemudian tidur	10	10	0	0%
Rata-rata akurasi				10%

mana *volunteer* terjatuh bebas ke depan pada matras dengan posisi awal tegak berdiri, pada kondisi ini modul *accelerometer* dapat mendeteksi semua kondisi jatuh yang terjadi. Kondisi kedua adalah kondisi dimana *volunteer* terjatuh bebas ke samping pada matras dengan posisi awal tegak berdiri, pada kondisi ini modul *accelerometer* dapat mendeteksi 9 kondisi jatuh dari 10 kali pengujian. Kondisi terakhir adalah kondisi dimana *volunteer* terjatuh bebas ke belakang pada matras dengan posisi awal tegak berdiri, pada kondisi ini modul *accelerometer* dapat mendeteksi semua kondisi jatuh yang terjadi.

Untuk kasus-kasus seperti *false positive*, modul *accelerometer* memiliki tingkat akurasi yang kurang baik dalam pendeteksian jatuh di beberapa kondisi. Contohnya pada kondisi berdiri kemudian duduk, hasil pendeteksian kondisi jatuh memiliki akurasi sebesar 20%. Sedangkan pada kondisi melompat lalu tidur, modul *accelerometer* mengatakan kondisi tersebut sebagai jatuh. Dalam hal ini, kondisi kedua hampir mustahil untuk dilakukan manula, namun penulis ingin menunjukkan bahwa ada beberapa kondisi yang menyerupai kondisi jatuh namun sebenarnya bukan jatuh.

4.1.2 Perhitungan *Beats per Minute*

Pada poin ini akan dilakukan pengujian untuk membandingkan tingkat akurasi perhitungan *BPM* menggunakan modul *ECG ad8232* terhadap perhitungan manual. Pengujian dilakukan sebanyak sepuluh kali dengan lama pengujian adalah satu menit. Hasil

Tabel 4.3: Tabel pengujian Modul *ad8232*

Pengujian ke	Hitung modul	Hitung manual	Galat
1	64 bpm	72 bpm	11.12%
2	60 bpm	71 bpm	15.49%
3	64 bpm	71 bpm	9.86%
4	58 bpm	70 bpm	17.14%
5	62 bpm	69 bpm	10.14%
6	65 bpm	73 bpm	10.96%
7	70 bpm	75 bpm	7.14%
8	72 bpm	74 bpm	8.78%
9	59 bpm	73 bpm	19.17%
10	68 bpm	74 bpm	8.11%
Rata-rata			11.80%

dari pengujian ditunjukkan pada tabel 4.3

4.1.3 Keberhasilan Pengiriman Data

Pada poin ini, akan diuji seberapa banyak data yang berhasil dikirim dari *wearable device* menuju *gateway* sampai data tersimpan di *database*. Pengujian ini dilakukan sebanyak sepuluh kali pengiriman data dari modul *ESP8266* menggunakan jaringan lokal dengan *gateway* menggunakan *repeater TP_LINK WA830RE*. Hasil pengujian keberhasilan pengiriman ditunjukkan pada tabel 4.4. Kemudian diuji pula waktu pengiriman yang terjadi dari *wearable device* menuju *gateway* sampai *script GET* dijalankan oleh *gateway*. Ada waktu yang dibutuhkan dari proses pemanggilan *script GET* hingga penyimpanan *database*, namun karena penyimpanan bersifat lokal, waktu yang dibutuhkan sangat sedikit sekali sehingga penulis memutuskan untuk mengabaikan waktu pada kondisi tersebut.

Dari sepuluh kali pengiriman, *wearable device* berhasil untuk mengirim semua data tersebut dan data diterima dengan baik sampai pada *database gateway*. Hasil ini menunjukkan keandalan *gateway* yang telah dikonfigurasi serta keandalan modul *ESP8266* dalam pengiriman data dalam jaringan lokal.

Tabel 4.4: Tabel pengujian pengiriman data menuju *gateway*

Pengiriman ke	Status Pengiriman	Response Pengiriman
1	Berhasil	200 (OK)
2	Berhasil	200 (OK)
3	Berhasil	200 (OK)
4	Berhasil	200 (OK)
5	Berhasil	200 (OK)
6	Berhasil	200 (OK)
7	Berhasil	200 (OK)
8	Berhasil	200 (OK)
9	Berhasil	200 (OK)
10	Berhasil	200 (OK)

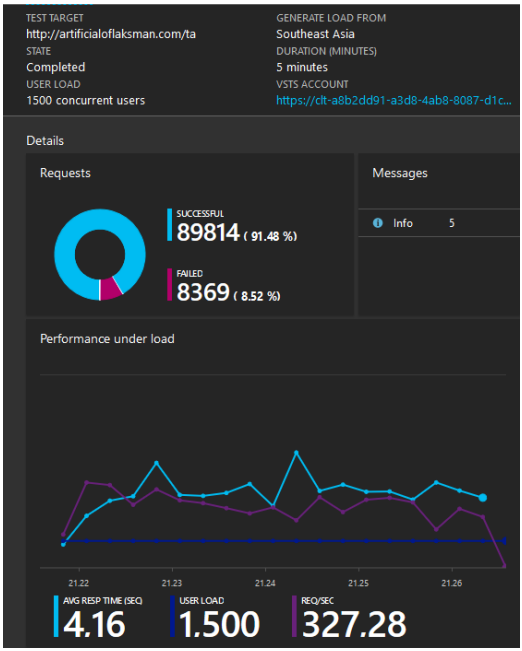
4.2 Pengujian *Cloud Server*

Dalam pengujian kali ini, akan diuji seberapa handal *server* dalam menerima *request* dari *user* untuk memantau kondisi manula. *Request* dapat berupa pemantauan melalui LINE *messenger* ataupun membuka aplikasi pemantau karena dua *service* tersebut berjalan pada satu *server*. Pengujian yang dilakukan kali ini adalah mencoba melakukan *request* terhadap page pemantau dalam waktu lima menit dengan jumlah *user* yang bervarian. Hasil pengujian disajikan seperti yang ditunjukkan pada tabel 4.5 dan gambar 4.2.

Tabel 4.5: Tabel pengujian *cloud server*

Jumlah user	Req. Sukses	Req. Gagal	Requset/sec
100	79943	0	266.48
200	94579	0	315.26
500	93661	0	312.2
1000	99866	0	332.89
1500	89814	8369	327.28
2000	100989	10183	352.93

Dari hasil pengujian terhadap *cloud server*, *cloud server* dapat menerima *request* dalam satu waktu hingga seribu *user*. Dari data terlihat jika seribu user melakukan request dalam waktu yang sama dalam kurun waktu lima menit, server masih bisa menerima



Gambar 4.2: Contoh *stress test* pada *cloud server*

semua *request* dan mengirim semua respon kembali kepada *user*. Sedangkan jika beban *user* dinaikkan menjadi seribu lima ratus *user*, *server* mengalami kegagalan dalam menerima *request* tetapi masih bisa melayani *user* dengan waktu rata-rata respon yang baik. Hal ini menunjukkan *server* bisa dikatakan handal karena dapat melayani lebih dari seribu *user* dalam satu waktu, selain itu juga hal ini membuktikan bahwa *page* yang dibuat untuk memantau kondisi serta aktivitas manula terbilang tidak terlalu membebani *resource server*. Pada gambar 4.2 ditunjukkan pengujian saat melakukan *stress test* pada *cloud server*.

4.2.1 Uji Sistem Notifikasi

Pada poin ini, pengujian dilakukan dengan mencoba melakukan *trigger* sebanyak sepuluh kali pada *server* untuk mengirim no-

Tabel 4.6: Tabel pengujian waktu pengiriman notifikasi

Trigger ke	Status Penerimaan	Interval Waktu
1	Diterima	9.1 detik
2	Diterima	5.8 detik
3	Diterima	6.1 detik
4	Diterima	5.5 detik
5	Diterima	5.8 detik
6	Diterima	6.0 detik
7	Diterima	5.6 detik
8	Diterima	5.2 detik
9	Diterima	5.3 detik
10	Diterima	5.4 detik
Rata-rata		5.98 detik

tifikasi kepada *LINE messenger* kerabat manula. Hal yang diuji pada poin ini adalah waktu yang dibutuhkan notifikasi untuk sampai kepada *smartphone* pengguna dari awal *trigger* dimulai. Selain itu, hal lainnya yang diperhatikan adalah apakah notifikasi diterima oleh *user* atau tidak. Hasil pengujian pada poin ini dapat dilihat pada tabel 4.6. Sedangkan untuk contoh notifikasi yang diterima oleh *user* dapat dilihat pada gambar 4.3.

Dari hasil pengujian trigger notifikasi, waktu rata-rata yang dibutuhkan untuk menyatukan informasi serta mengirimnya menjadi notifikasi sekitar 5.98 detik. Dari data yang diberikan juga dapat ditarik analisa, untuk *trigger* pertama terjadi lebih lama dibandingkan dengan *trigger* selanjutnya, ini terjadi dikarenakan *server* belum menyimpan *cache* notifikasi untuk mengirim notifikasi kepada *user*, namun setelah *server* menyimpan *cache*, notifikasi yang dikirim menjadi jauh lebih cepat dibanding dengan pertama kali notifikasi dikirim.

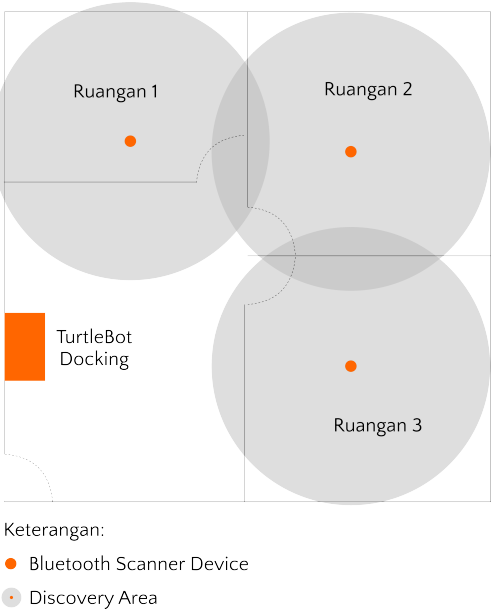
4.3 Pengujian *Gateway*

Sebelum melakukan pengujian *gateway*, ada beberapa hal yang perlu diperhatikan, pertama adalah memastikann bahwa *web-server*

Gambar 4.3: Contoh notifikasi yang diterima

telah dinyalakan dan *node.js* sebagai tunnel juga sudah dinyalakan. Setelah semua *service* sebagai *server* telah dinyalakan maka selanjutnya adalah menyalakan *robot* melalui *script* pada terminal agar *robot* selalu dalam keadaan *listen* untuk menerima sinyal dari *wearable device*.

Selanjutnya adalah menyiapkan desain denah untuk diterapkan menjadi sebuah denah sederhana pada kenyataan. Desain denah memiliki beberapa informasi dimana diantaranya adalah ruangan, *discovery device*, *turtlebot docking* serta area yang menunjukkan jarak jangkau *discovery device*. Desain denah ini ditujukan untuk menguji sistem yang telah dibuat terutama pada *path planning* pa-



Gambar 4.4: Desain denah untuk pengujian

da *robot* untuk mengambil foto manula serta uji coba penentuan lokasi yang ditentukan oleh cubeacon. Desain denah yang digunakan pada tugas akhir ini ditunjukkan oleh gambar 4.4.

4.3.1 Pengujian Respon *Gateway*

Dalam pengujian respon *gateway*, *wearable device* akan mengirim data kondisi manula menuju *gateway* sebanyak sepuluh data kondisi secara berturut-turut untuk dilihat apakah *gateway* dapat menerima data tersebut dan menyimpannya menuju *local storage*. Hasil dari pengujian ini ditunjukkan oleh tabel 4.7. Dari hasil didapatkan bahwa penerimaan data dari *wearable device* dapat diterima secara utuh oleh *gateway*. Hal ini menunjukkan bahwa *gateway* tidak mengalami penurunan *resource* walaupun telah dibebankan terhadap *service middleware* berupa *Robot Operating System* untuk menjalankan fungsi *robot* serta tidak terbeban terhadap *web-server*

yang sedang berjalan serta tidak terbeban dengan aktifnya *tunnel* untuk melakukan *forward* data menuju *cloud server*.

Tabel 4.7: Tabel pengujian penerimaan data dari *wearable device*

Penerimaan ke	Status Penerimaan	Status Koneksi
1	Berhasil	OPEN
2	Berhasil	OPEN
3	Berhasil	OPEN
4	Berhasil	OPEN
5	Berhasil	OPEN
6	Berhasil	OPEN
7	Berhasil	OPEN
8	Berhasil	OPEN
9	Berhasil	OPEN
10	Berhasil	OPEN

4.3.2 Pengujian Penentuan Lokasi

Pengujian pada poin ini dilakukan dengan menempatkan tiga *discovery device* pada tiga lokasi yang berbeda sesuai dengan denah yang diberikan sebelumnya. Pada pengujian ini, aplikasi *discovery* akan dipasang pada tiga *smartphone* dengan tipe yang sama kemudian dari ketiga *device* ini akan diambil data kekuatan sinyalnya terhadap *wearable device* untuk ditentukan lokasi manula berada. Pengujian ini menunjukkan bahwa dari sembilan kali pengujian, aplikasi *discovery* dapat menentukan posisi manula dengan baik dengan syarat *smartphone* harus *line-of-sight* dengan *wearable device*.

Pada pengujian ini lokasi smartphone pertama berada pada ruangan pertama, lokasi *smartphone* kedua berada pada ruangan kedua, dan lokasi *smartphone* ketiga berada pada ruangan ketiga. Sinyal *dBm* paling besar yang didapatkan *smartphone* merupakan lokasi manula saat itu. Posisi orang yang mengenakan *wearable device* akan berpindah-pindah tetapi tetap dalam satu ruangan yang sedang diuji. Pengujian lebih detil dapat dilihat pada tabel 4.8, pada tabel tersebut kolom pengujian ruangan menunjukkan posi-

Tabel 4.8: Tabel pengujian penentuan lokasi

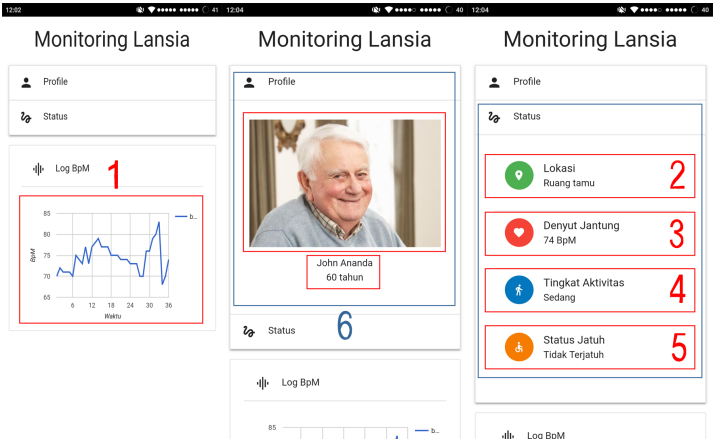
Pengujian ruangan	*Device 1*	*Device 2*	*Device 3*	Posisi didapat
Ruangan 1	-34 dBm	-75 dBm	-80 dBm	Ruangan 1
Ruangan 1	-40 dBm	-72 dBm	-79 dBm	Ruangan 1
Ruangan 1	-46 dBm	-70 dBm	-79 dBm	Ruangan 1
Ruangan 2	-69 dBm	-38 dBm	-71 dBm	Ruangan 2
Ruangan 2	-68 dBm	-36 dBm	-65 dBm	Ruangan 2
Ruangan 2	-70 dBm	-48 dBm	-75 dBm	Ruangan 2
Ruangan 3	-75 dBm	-70 dBm	-35 dBm	Ruangan 3
Ruangan 3	-73 dBm	-68 dBm	-37 dBm	Ruangan 3
Ruangan 3	-70 dBm	-67 dBm	-39 dBm	Ruangan 3

si *wearable device* yang sedang dideteksi, kolom *device* merupakan kekuatan sinyal yang didapat pada setiap *smartphone* dan posisi didapat adalah posisi yang didapat setelah diproses oleh *server*.

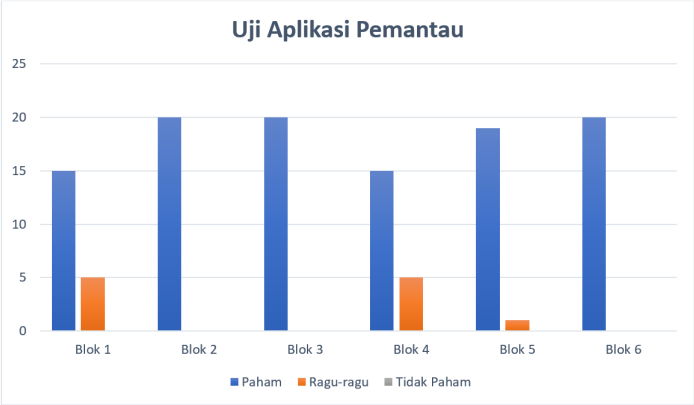
4.4 Pengujian Aplikasi Pemantau

Pengujian pada poin ini dititikberatkan pada *user experience*, sehingga pada pengujian ini akan dilakukan *survey* mengenai pengalaman responden dalam menggunakan aplikasi pemantau baik melalui *chat room* ataupun melalui aplikasi berbasis *web*. Pada aplikasi pemantau berbasis *web*, Aplikasi memiliki beberapa komponen didalamnya diantaranya adalah *profile*, *status* dan *log BPM*. *profile* berisi foto yang dapat di *pop up*, nama, dan umur dari manula yang mengenakan *device*, status berisi beberapa kondisi yang dapat dipantau oleh *device* seperti lokasi, denyut jantung, tingkat aktivitas dan status jatuh, kemudian *log BPM* berisi grafik dari denyut jantung manula dalam satu hari dan akan diganti per hari. *User* akan diminta untuk memberi penilaian dengan skala 0-5 untuk menguji keyakinan *user* terhadap fungsi dari masing-masing menu. Selain itu *user* juga akan diminta menilai aplikasi dari segi kenyamanan penggunaan aplikasi .

Aplikasi dinilai oleh beberapa user melalui sebuah link yang telah diberikan. Kemudian pengguna akan diminta untuk menguji aplikasi dari segi pemahaman, kenyamanan dan tampilan terhadap aplikasi. Dari 20 responden, sebanyak 20 responden merasa paham terhadap informasi yang ditampilkan pada blok 2, 3, dan 6. Se-



Gambar 4.5: Blok pada elemen *user interface* yang diujikan



Gambar 4.6: Grafik pengujian aplikasi pemantau

dangkan sebanyak 15 responden merasa paham dan 5 responden merasa ragu-ragu terhadap informasi yang ditampilkan pada blok 1 dan 4. Terakhir, sebanyak 19 responden merasa paham dan 1 responden merasa ragu-ragu terhadap informasi yang ditampilkan pada blok 5.

Tabel 4.9: Tabel pengujian jarak terhadap kekuatan sinyal

Jarak	***Received Signal Strength***	**Status sinyal**
5 cm	-31dBm	Sangat Baik
10 cm	-34dBm	Sangat Baik
20 cm	-41dBm	Sangat Baik
30 cm	-46dBm	Sangat Baik
40 cm	-55dBm	Sangat Baik
80 cm	-71dBm	Cukup
160 cm	-81dBm	Tidak Baik
200 cm	-88dBm	Tidak Baik

4.5 Pengujian Aplikasi *Discovery Device*

Pengujian pada aplikasi *Discovery Device* ditujukan untuk mengetahui hubungan jarak terhadap *Received Signal Strength Indication (RSSI)*. Semakin jauh jarak cubeacon terhadap *discovery device*, semakin kecil pula *RSSI* yang diterima oleh *discovery device*. *Signal Strength* dikatakan sangat baik jika sinyal yang diterima memiliki kekuatan hingga -30dBm. Sinyal dikatakan baik jika sinyal yang diterima memiliki kekuatan hingga -67dBm. Sinyal dikatakan cukup jika sinyal yang diterima memiliki kekuatan hingga -70dBm. Sinyal dikatakan tidak baik jika sinyal yang diterima memiliki kekuatan hingga -80dBm. Sinyal dikatakan buruk jika sinyal yang diterima memiliki kekuatan hingga -90dBm. [17]

Pengujian ini menggunakan *Smartphone Xiaomi MI4i* sebagai *discovery device* dan akan diuji jarak terjauh dalam pendeteksian cubeacon. Hasil pengujian ditunjukkan pada tabel 4.9.

4.6 Survey Kebutuhan Sistem serta Kenyamanan Device

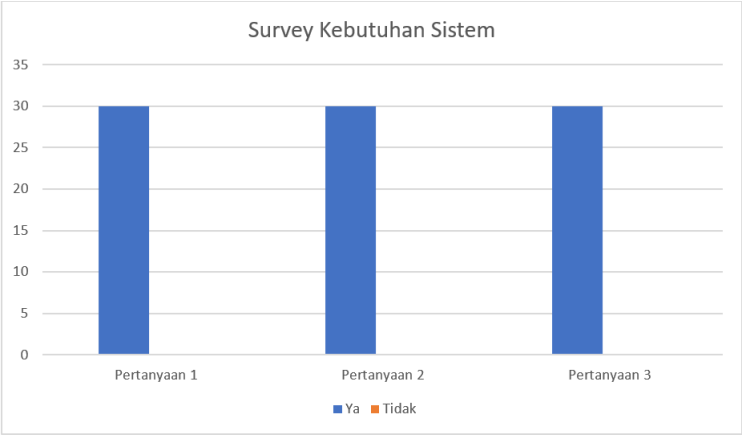
Survey pada poin ini bertujuan untuk mengetahui apakah sistem yang dikembangkan akan dibutuhkan pada masa mendatang dan untuk mengetahui seberapa nyaman *device* yang sudah dikembangkan saat digunakan oleh manula.

4.6.1 Survey Kebutuhan Sistem di Masa Mendatang

Survey dilakukan dengan menyebar kuesioner secara *online* dengan *target* orang yang berusia produktif dan memiliki orangtua. Diberikan sebanyak tiga pertanyaan mengenai sistem yang dikembangkan dalam tugas akhir ini. Pertanyaan yang diberikan adalah sebagai berikut:

1. Apakah anda merasa khawatir jika anda meninggalkan orangtua anda sendiri dirumah ? (misal akan terjatuh)
2. Apakah anda membutuhkan sistem seperti yang dijelaskan diatas untuk memantau kondisi orangtua anda ?
3. Apakah anda menyarankan agar sistem ini nantinya diterapkan di Indonesia ?

Seperti yang ditunjukkan pada gambar 4.7, dari tiga puluh responden, semua responden mengatakan iya untuk setiap pertanyaan yang diberikan. Ini menunjukkan bahwa sistem ini sangat dibutuhkan oleh masyarakat Indonesia.



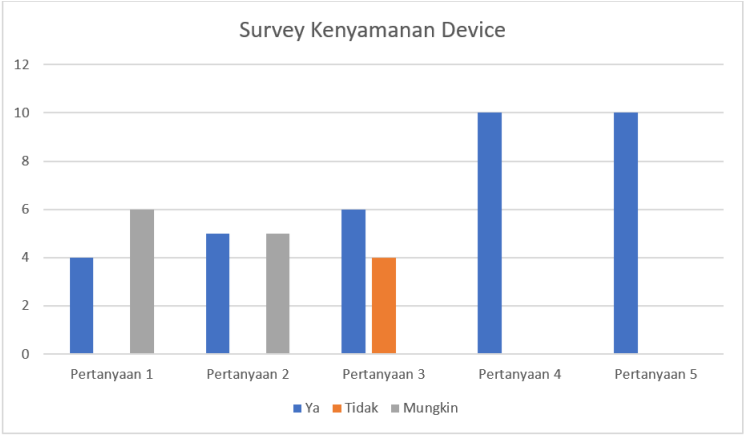
Gambar 4.7: Survey kebutuhan akan sistem

4.6.2 Survey Kenyamanan Penggunaan Device

Survey pada poin ini bertujuan untuk mengetahui apakah *device* yang dikembangkan sudah nyaman untuk digunakan oleh manula dan apakah sistem ini disarankan untuk diterapkan di Indonesia. *Survey* dilakukan dengan melakukan *survey* secara langsung kepada manula atau orang dengan usia 55 tahun keatas. Diberikan sebanyak lima pertanyaan mengenai sistem yang dikembangkan dalam tugas akhir ini. Pertanyaan yang diberikan adalah sebagai berikut:

1. Apakah anda membutuhkan sebuah sistem seperti yang sudah dijelaskan diatas ?
2. apakah anda nyaman jika mengenakan device ini di pinggang ?
3. Apakah desain dari alat sudah baik ?
4. Apakah anda terbantu jika alat ini dapat memberikan informasi kondisi vital anda kepada keluarga anda melalui smartphone ?
5. Apakah anda menyarankan untuk alat ini digunakan secara massal ?

Seperti yang ditunjukkan pada gambar 4.8, dari sepuluh responden, empat responden mengatakan iya dan enam responden



Gambar 4.8: Survey kenyamanan *device*

mengatakan mungkin pada pertanyaan pertama. Lima responden mengatakan iya dan lima responden mengatakan mungkin pada pertanyaan kedua. Enam responden mengatakan sudah dan empat responden mengatakan belum pada pertanyaan ketiga. Sepuluh responden mengatakan iya pada pertanyaan keempat. Sepuluh responden mengatakan iya pada pertanyaan kelima.

BAB 5
PENUTUP

5.1 Kesimpulan

Dari pelaksanaan dan pengujian sistem yang sudah dilakukan, penulis berhasil mengimplementasikan *Internet of Things* sebagai pemantau kondisi manula dengan menggunakan *wearable device* sebagai pengakuisisi data kondisinya serta menyandingkan *modern messenger* sebagai sistem notifikasinya. Kemudian untuk lebih detil dapat ditarik beberapa kesimpulan sebagai berikut:

1. Penelitian ini membuktikan bahwa *Internet of Things* bisa disandingkan dengan *robot* yang berjalan pada sistem operasi *robot* sebagai *middleware gateway*, yang kemudian dapat dibuat suatu sistem yang lebih besar bernama *Internet of Robot Things*.
2. Modul *accelerometer* sudah cukup baik dalam mendeteksi jatuh bebas pada manula terbukti dengan tingkat keberhasilan pendeteksian adalah 96%. Sedangkan untuk pendeteksian gerakan jatuh yang lebih rumit, modul accelerometer masih kurang mumpuni untuk melakukan pendeteksian dibuktikan dengan tingkat keberhasilan pendeteksian hanya 10%. Kemudian untuk modul ECG, modul dikatakan cukup baik dalam menghitung *beats per minute* dari manula ditunjukkan dengan tingkat akurasi modul dalam pendeteksian sebesar 88,2% namun tidak disarankan untuk melakukan diagnosa penyakit.
3. *User interface* aplikasi pemantau dapat mudah dipahami oleh pengguna ditunjukkan dengan hasil *survey* sebanyak 90,83% pengguna mengatakan paham dengan elemen-elemen yang terdapat pada aplikasi pemantau.
4. Dari *survey* yang telah dilakukan, baik manula maupun pengguna aplikasi pemantau sama-sama menyarankan agar sistem ini diterapkan dalam kehidupan. Hal ini berarti masyarakat sangat membutuhkan sistem pemantauan ini didalam kehidupan mereka.

5.2 Saran

Demi pengembangan lebih lanjut mengenai tugas akhir ini, disarankan beberapa langkah lanjutan sebagai berikut :

1. Menambahkan sensor yang mungkin dapat memantau kondisi manula atau menambah keefektifan sensor sebelumnya dalam pendeteksian jatuh atau denyut jantung.
2. Menambahkan fitur pada *robot* yang dapat dikontrol melalui *wearable device* manula.
3. Meningkatkan *resources* pada *server* sehingga dapat melayani lebih banyak *user*.

DAFTAR PUSTAKA

[1] Statistik Penduduk Lanjut Usia 2014. Badan Pusat Statistik, 2015. (Dikutip pada halaman xi, 1, 2).

[2] "Turtlebot2." http://www.turtlebot.com/turtlebot2/. Diakses pada: 2017-11-30. (Dikutip pada halaman xi, 5).

[3] sparkfun, Small, Low Power, 3-Axis 3 g Accelerometer. (Dikutip pada halaman xi, 9).

[4] "Target heart rates." http://www.heart.org/HEARTORG/HealthyLiving/PhysicalActivity/FitnessBasics/Target-Heart-Rates_UCM_434341_Article.jsp#.WkJrrjeySUk. Diakses pada: 2017-11-30. (Dikutip pada halaman xi, 10).

[5] "How do ibeacons work?." https://www.javacodegeeks.com/2014/01/how-do-ibeacons-work.html. Diakses pada: 2018-01-14. (Dikutip pada halaman xi, 11).

[6] "ibeacon - beacon scanner." https://codeday.me/news/20170530/19671.html. Diakses pada: 2018-01-14. (Dikutip pada halaman xi, 12).

[7] "Important facts about falls." https://www.cdc.gov/homeandrecreationalsafety/falls/adultfalls.html. Diakses pada: 2018-01-09. (Dikutip pada halaman xi, 18).

[8] "About ros." http://www.ros.org/about-ros/. Diakses pada: 2018-01-14. (Dikutip pada halaman 6, 7).

[9] "Berkenalan dengan arduino nano." http://ecadio.com/mengenal-dan-belajar-arduino-nano. Diakses pada: 2018-01-14. (Dikutip pada halaman 7).

[10] "About - ifttt." https://ifttt.com/about. Diakses pada: 2018-01-14. (Dikutip pada halaman 12).

[11] V. Beal, "What is cloud storage?." www.webopedia.com/TERM/C/cloud_storage.html. Diakses pada: 2017-09-30. (Dikutip pada halaman 13).

[12] M. Borgmann, T. Hahn, M. Herfert, T. Kunz, M. Richter, U. Viebeg, and S. Vowe, On the Security of Cloud Storage Services. Fraunhofer Institute for Secure Information Technology SIT, Germany: Mediendienstleistungen des, Fraunhofer-Informationszentrum Raum und Bau IRB, Stuttgart, 2012. (Dikutip pada halaman 14).

[13] "In introduction to http basics." https://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html. Diakses pada: 2017-10-29. (Dikutip pada halaman 14).

[14] T. S. et al, "Fall detection algorithm based on trixial accelerometer and magnetometer," in Engineering Letters, 24:2. (Dikutip pada halaman 31).

[15] Y. Z. Falin Wu, Hengyang Zhao and H. Zhong, "Development of a wearable-sensor-based fall detection system," International Journal of Telemedicine and Applications, vol. vol. 2015, p. 11, 2015. (Dikutip pada halaman 32).

[16] S. B. K. et al., "Fall detection using single tri-axial accelerometer," in ASEE 2014 Zone 1 Conference, 2014. (Dikutip pada halaman 32).

[17] "Understanding rssi." https://www.metageek.com/training/resources/understanding-rssi.html. Diakses pada: 2017-11-30. (Dikutip pada halaman 55).

BIOGRAFI

Anak Agung Ngurah Surya Laksamana, lahir pada 11 Februari 1996 di Pematang Siantar, Sumatera Utara. Penulis lulus dari SMP Negeri 3 Batam pada tahun 2011 kemudian melanjutkan pendidikan ke SMA Negeri 8 Bandung hingga akhirnya lulus pada tahun 2014. Penulis kemudian melanjutkan pendidikan ke S1 Departemen Teknik Komputer FTE-ITS Surabaya Bidang Studi Telematika. Saat di kuliah penulis aktif menjadi Asisten laboratorium B401 Komputasi Multimedia dan pernah menjabat sebagai koordinator dalam bidang *Web Development*. Selama masa kuliah, penulis aktif di Microsoft sebagai *Microsoft Student Partner* Indonesia dalam membagi keilmuan seputar teknologi kepada mahasiswa kampus, selain itu penulis aktif mengikuti ajang serta kompetisi teknologi tingkat mahasiswa ataupun umum. Penulis banyak menghabiskan waktu untuk mengeksplor isu-isu terkait teknologi dunia. Untuk kepentingan kepada penulis, dapat menghubungi agunable@gmail.com

Halaman ini sengaja dikosongkan